

Symmetries of Living Systems

Symmetry Fibrations and Synchronization in Biological Networks

HERNÁN A. MAKSE, PAOLO BOLDI,
FRANCESCO SORRENTINO, IAN STEWART

Cambridge University Press, 2025

Preface

A symmetry is a ‘change without change’. As simple as it sounds, this concept is the fundamental cornerstone that unifies all branches of theoretical physics. Virtually all physical laws—ranging from classical mechanics and electrodynamics to relativity, quantum mechanics, and the standard model—can be expressed in terms of symmetry invariances. In this book, we explore whether the same principle can also explain the emergent laws of biological systems.

We introduce a new geometry for biological networks and AI architectures, drawing inspiration from the mystic genius of Grothendieck’s fibrations in category theory. We attempt to bridge the gap between physics and biology using symmetries but with a twist. The traditional symmetry groups of physics are global and too rigid to describe biology. Instead, the novel notion of symmetry fibration is local, flexible, and adaptable to evolutionary pressures, providing the right framework for understanding biological complexity.

Specifically, networks for which there are transformations of nodes that leave input trees invariant create functionally coherent dynamics. In other words, this more general symmetry invariance is necessary and sufficient to ensure that a given biological network configuration can support a synchronized function. In this book, we review the theoretical progress over the last decades from mathematics, physics, computer science, dynamical systems, and graph theory that has led to the discovery of symmetry fibrations in biological networks.

These symmetries act as organizing principles for biological networks. They serve as effective tools for describing the structure of these networks, blending geometry and topology. Fibrations explain how structure dictates function across various biological domains, including the transcriptome, proteome, metabolome, and connectome. Additionally, they facilitate a reduction in the dimensionality of the network, simplifying it into its fundamental building blocks for biological computation.

This book is an attempt to describe, in simple terms, the branches of theoretical physics and mathematics that are relevant to the application of symmetries to biology and to make these ideas accessible to a wider audience of researchers. We provide details for numerous biological networks of current interest.

Organization of the book

Part I introduces the theoretical concepts needed to understand symmetry fibrations. We do this through simple examples, either mathematical or biological. From Chapter 3 onwards, we also present formal definitions of these concepts. The aim is to make the material accessible to non-mathematicians while avoiding over-simplification. Part II demonstrates,

in much greater detail, exactly how these concepts apply to a variety of biological networks ranging from genetic networks to the brain.

Part I begins with Chapter 1, which poses the main question of this book: can the same symmetry principles that describe physical systems be applied to biological systems? We argue that a more general symmetry notion is required to take account of the flexibility found in biology.

Chapter 2 gives an informal overview of the essential ideas involved in this more general kind of symmetry: symmetry fibrations, input trees, balanced colorings, and synchrony patterns. We also discuss admissible equations, which are the most general ordinary differential equations that are compatible with the network topology and, therefore, include all possible model equations for the network concerned. We explain the difference between the global group-theoretic symmetries used in physics and the local symmetries of fibrations, which we suggest are more suitable for biology.

Chapter 3 starts by defining basic concepts such as graphs, networks, partitions, complete versus cluster synchronization, and elaborates on a recurrent topic in this book: the structure-function relation. We discuss how symmetry fibrations guarantee the synchronization of network node activity. The sets of synchronized nodes are called fibers, and they represent nodes in the network that have the same dynamical state, i.e., they are synchronized. This chapter includes the mathematical proof of how symmetry fibrations lead to synchronization in fibers.

Chapter 4 describes automorphisms and their relation to cluster synchronization. These symmetries describe all physical systems but few biological ones. However, they serve as an introduction to the biological symmetry fibrations to be discussed in Chapters 5 and 6.

Chapter 5 describes the graph fibration formalism, including the definition of input-tree isomorphisms and minimal graph fibrations, leading to symmetry fibrations, balanced colorings, and the equivalent concept of an equitable partition. These are the natural symmetries of biological networks. We discuss necessary conditions for synchrony to be possible and relate these to input trees.

Chapter 6 expands on the notion of a fibration symmetry, introduced informally in Chapter 2. We begin with the more general notion of a graph homomorphism, and then specialize to fibrations. We relate fibrations to equitable partitions, and emphasize the ‘lifting property’, which distinguishes fibrations from homomorphisms. The connections between fibrations, synchrony, balanced colorings, and equitable partitions are examined in greater depth.

Chapter 7 is mainly about the groupoid formalism, a mathematical context suited to the local symmetries that arise from fibrations. A groupoid is like a group, except that a groupoid need not be closed under the operation of composition. We give a brief, concrete discussion of groupoids that avoids most technicalities.

Chapter 8 focuses on the important issue of stability. A dynamical state is stable if it remains essentially unchanged after a sufficiently small perturbation is applied. There are several stability notions, and we focus on the main ones. We discuss the stability of equilibria and apply the theory to some biological examples. The chapter ends with a brief introduction to bifurcations, where the stability of a state changes as parameters are varied, leading to radical changes in the dynamics.

Chapter 9 generalizes the setting from graphs to related structures, such as multi-layered

and weighted graphs. In particular, we discuss hypergraphs, which formalize specific types of many-body interaction. These types of network are important for a more complete description of biological networks across the spectrum of biological interactions from gene regulation, protein-protein interactions, protein-metabolite interactions and metabolism.

Chapter 10 discusses the relation between topological fibrations, graph fibrations and fiber bundles. Fiber bundles are at the foundations of particle physics as the formalism used to describe the symmetries of the standard model. This chapter attempts to make a connection between the formalism of fibrations in the broader realm of theoretical physics.

Chapter 11 discusses how fibrations provide a structure for biological networks that is robust to failure of its components, e.g., under gene mutations. A comparison with group symmetry structures shows that networks with fibrations are less vulnerable to such failures than those with group symmetries. This suggests an evolutionary drive for fibrations in biological networks. We discuss how structure can constrain behavior (phenotype).

Chapter 12 proposes a geometrization of biological networks, by analogy with the geometrization of physics, by following a parallel between fibrations in biology and fiber bundles in physics. Using a financial analogue proposed by Maldacena we show that the concepts of curvature and connection in physical fiber bundles have analogues in biological fibrations, such as those in genetic networks and the brain.

Chapter 13 discusses algorithms to find fibers in networks by exploiting the link between fibers and balanced colorings. A set of algorithms and their implementations is provided to calculate fibrations and fiber building blocks easily for any network. Online sources for algorithms used in this book are listed in Appendix A.

Chapter 14 starts Part II. We show how symmetry fibrations describe a hierarchy of fiber circuits across biological networks, especially transcription regulatory networks, but also more generally. We describe how these fibers define the building blocks of these networks and provide a broad classification of building blocks.

Chapter 15 applies the ideas of Chapter 14 to specific simple fibration building blocks of biological significance. We return to some of these networks later, in greater detail and with more realistic models. The circuits discussed include the autoregulation loop, several types of feed-forward fiber, Fibonacci fibers, and binary and n -ary trees. We discuss operons and regulons, which can be viewed as trivial examples of fibers, and analyze how building blocks vary across different species.

Chapter 16 considers metabolic networks and enzyme networks, where more complex building blocks arise. We focus on *E. coli* where, in particular, complex composite Fibonacci building blocks appear. These circuits involve longer-range feed-forward and feedback loops.

Chapter 17 compares the building blocks obtained from fibrations with network motifs and modules, the popular ways to identify building blocks of biological networks. We define p -values and Z-scores, which quantify the concepts involved. A key difference between motifs and fibration building blocks is that motifs are defined statistically, whereas fibration building blocks are defined dynamically.

Chapter 18 discusses the cell as a computational device. It shows how fibration symmetry breaking can identify genetic circuits analogous to toggle switches and memory storage flip-flops that build the computational logic machinery of genetic networks. This leads to

the identification of the function of each gene in a genetic network of bacteria as a part of a computational logic structure of the network, which is considered as a finite state machine of computation.

Chapter 19 concerns the key issue of complexity reduction. Most real biological systems are inordinately complex; to understand them we must somehow reduce the complexity. We show that the application of fibrations to genetic networks leads to a systematic method for uncovering a minimal functional network, thus reducing biological complexity to its minimal form. We analyze some biological examples in detail.

Chapter 20 describes applications of fibrations to biological systems displaying cluster synchronization such as gene coexpression and neural synchronization. The main aim is to investigate the structure \rightsquigarrow function relation. We also describe the types of systems of admissible dynamical equations that can be studied by fibrations.

Chapter 21 asks how living organisms can exist when everything in biology conspires against synchrony. We attempt to answer this question in terms of weak symmetry breaking, in which idealized symmetric networks are modified by natural selection to break symmetry while preserving approximate synchrony. In essence, this chapter is about the relationship between idealized mathematical models and realistic biology.

Chapter 22 addresses the issue of reconstructing idealized networks from incomplete biological information. This is the converse of the structure \rightsquigarrow function relation: the function \rightsquigarrow structure relation.

Chapter 23 asks whether it is realistic to seek organizing principles for brains. In this chapter, we consider circuits in the *C. elegans* connectome. We apply the methods developed in previous chapters to use fibrations to classify cell types and consider locomotion in *C. elegans* from this point of view.

Chapter 24 continues the analysis of brains in the more complex case of the mouse. Here, we focus on memory storage and the notion of an engram.

Chapter 25 pushes ahead into the even more complex structures of the human brain, with an emphasis on language. The role of synchrony is emphasized, and we discuss fibration symmetry breaking in the human language connectome.

Chapter 26 discusses fibration symmetries in artificial intelligence, from the general viewpoint of Geometric Deep Learning and Felix Klein's Erlangen Program, which unified geometry in the 1870s.

Chapter 27 ends the book with an overview of its main messages and an outlook for future work.

Appendix A then provides the code and data to perform fibration data analysis and reproduce all the results of the book.

How to read this book

This book aims to provide insights across various levels of complexity to inspire interdisciplinary collaboration:

- For readers new to the concepts of symmetries in networks and seek a quick yet informative overview, we recommend focusing on Chapters 1 and 2 only. These chapters

lay the foundation and discuss all concepts needed for the application of symmetries in biology.

- If you are interested in the mathematically rigorous formulation of fibrations, we encourage you to read Chapters 3 through 12, which provide a detailed examination of the mathematical principles that underpin the fibration formalism.
- For those with an interest in the practical implications of fibrations in both biological networks and artificial intelligence, transitioning to Part II after completing the first two introductory chapters is advisable. Each chapter includes self-contained real-world applications relevant to practical scenarios.
- Similarly, if your goal is to perform a fibration/symmetry analysis of complex networks, you can directly proceed to Chapter 13 following your review of the first two introductory chapters. Here, you will find the essential algorithms, along with the accompanying codes provided in the Appendix A, to facilitate practical applications.

Acknowledgment and fibration companion website

We are deeply indebted to the many enlightening discussions we have shared with our collaborators. Our journey into the fascinating world of fibrations in biology began with inspiring conversations with Flaviano Morone as we sought to develop a theory that bridges the gap between structure and function in biological networks. Throughout this journey, we have been fortunate to collaborate with an exceptional group of individuals who have deepened our understanding of symmetries in biology, including Luis Álvarez-García, Francesca Arese-Lucini, Pedro Augusto, Bryant Ávila, José Soares Andrade Jr, Pablo Balenzuela, Guido Caldarelli, Santiago Canals, Sofía Morena del Pozo, Andrea Gabrielli, Raquel García-Hernández, Tommaso Gili, Martin Golubitsky, Alireza Hashimi, Cecilia Ishida, Andrei Holodny, Ian Leifer, Wolfram Liebermeister, Higor Monteiro, Amir Nazerian, Lucas Parra, David Phillips, Saulo Reis, Nastassia Samadzelkava, Matteo Serafino, Mariano Sigman, Silvina Tomassone, Osvaldo Velarde, Sebastiano Vigna, Stefan Wuchty, and Manuel Zimmer.

The companion website of the book <https://fibration.org> includes resources, codes, and datasets discussed in this book. We would love to hear from our readers. Please send any comments or requests to hmakse@ccny.cuny.edu.

New York, USA
Milan, Italy
Albuquerque, USA
Coventry, UK

*Hernán A. Makse
Paolo Boldi
Francesco Sorrentino
Ian Stewart
February 2025*

Contents

Part I Theory: Symmetry Groups in Physics; Symmetry Fibrations in Biology	<i>page 1</i>
1 Biological Symmetry	3
1.1 Symmetry in physics and biology	3
1.2 Group-theoretic symmetries in biology	4
1.3 A more general notion of symmetry	8
1.4 Cluster synchronization and fibrations	9
2 Symmetry Fibrations in a Nutshell	13
2.1 Informal introduction to fibrations	13
2.2 Fibrations, input trees, balanced colorings, and synchrony	14
2.3 Symmetry fibration in <i>Escherichia coli</i>	15
2.3.1 Input trees	17
2.4 Admissible equations and cluster synchronization	18
2.5 Key principle	20
2.6 Global symmetry of automorphisms versus local symmetry of fibrations	20
2.7 Physics and biology	23
3 Graphs, Networks, and Synchronization	26
3.1 Basic definitions	26
3.2 Graphs, multigraphs, and hypergraphs	29
3.3 Setting up model equations	30
3.4 Admissible ODEs	31
3.4.1 From graph to admissible ODEs	31
3.4.2 From admissible ODEs to graph	33
3.5 Partition of a graph	35
3.6 Complete synchronization and cluster synchronization	36
3.6.1 Laplacian systems	37
3.6.2 Cluster synchronization and cellular function	38
3.7 From graph to network dynamics – from structure to function	39
4 Automorphisms and Symmetry Groups	41
4.1 Automorphisms and the symmetry group of a graph	41

4.1.1	Formal definitions	42
4.2	Cluster synchronization by automorphisms: orbital partition	44
4.2.1	Orbits define clusters	45
4.2.2	Orbital clusters and synchrony	46
4.2.3	Two types of quotient graph	48
5	Input Trees, Synchrony, Balanced Colorings, and Equitable Partitions	50
5.1	Intuition and motivation	50
5.2	Global symmetries versus local symmetries	52
5.2.1	Biology is robust	53
5.3	Modeling assumptions leading to synchrony	54
5.3.1	Necessary conditions for synchrony	56
5.4	Input sets and input trees	57
5.5	Examples of input trees	62
5.6	Input tree isomorphism	63
5.7	The graph isomorphism problem: testing input tree isomorphism	65
5.8	Minimal equitable partition and balanced coloring partition	67
5.8.1	Examples of balanced coloring partitions	72
5.9	Robust versus fragile synchrony	76
5.10	Comments on equivalent definitions in different disciplines	78
6	Graph Fibration Formalism	80
6.1	Graph homomorphisms	80
6.2	Graph fibrations	83
6.3	Examples of fibrations	83
6.4	Graph fibrations through equitable partitions	86
6.5	The lifting property	89
6.6	Hierarchy of symmetries from homomorphisms to fibrations (op-fibrations) to coverings to automorphisms	93
6.6.1	Preserving outputs instead of inputs: op-fibrations	94
6.6.2	A graph having different fibrations and op-fibrations	95
6.6.3	Preserving input and output at the same time: coverings	96
6.6.4	Levels of symmetry	97
6.7	Further thoughts on global versus local symmetries	99
6.8	Example of fibrations, synchrony, colorings, and balance	101
6.9	Fibers and minimal balanced colorings	103
6.10	Beyond the minimal balanced coloring	104
7	Symmetry Groupoid Formalism	107
7.1	Formulation in terms of symmetry groupoids	107
7.2	Groupoids	108
7.3	Input isomorphisms between input sets	110
7.4	Symmetry isomorphisms and symmetry groupoid	112
7.5	Example of groupoids and cluster synchronization	115

7.6	Algebraic formulation: lack of composition in groupoids	117
7.7	Relations between properties of fibrations, groups, and groupoids	119
7.8	Formal definition of a groupoid	120
7.9	Hierarchy of algebraic structures	122
8	Stability and Synchronizability	125
8.1	Biology needs stability	125
8.1.1	Existing work on stability of cluster states	127
8.2	Notions of stability	128
8.3	Stability of equilibria of linear ODEs	130
8.4	Stability of equilibria of nonlinear ODEs	132
8.5	Two examples	133
8.5.1	Synchronous and transverse eigenvalues and eigenvectors	137
8.6	Some small genetic circuits	138
8.7	Protein/mRNA network structure	139
8.7.1	Hill functions	140
8.7.2	Bipartite structure	140
8.7.3	Relation between GRN and PRN	141
8.7.4	Relation between Jacobian and adjacency matrices	141
8.7.5	Adjacency matrix and Jacobian	142
8.8	Stability analysis of biologically relevant circuits	144
8.8.1	Stability analysis for lock-on	144
8.8.2	Bistability of the toggle-switch	147
8.8.3	Comparison of metabolator and Smolen	151
8.9	Bifurcations—a short sketch and examples	153
8.9.1	First bifurcation	155
8.9.2	Hopf bifurcation in the symmetric repressilator	158
8.9.3	Discussion and Conclusions	161
8.10	Master stability function for complete synchronization	161
8.11	A master stability function approach for cluster synchronization	164
8.11.1	Examples of cluster stability analysis	167
8.12	Structural stability: changes to equations, parameters, and graphs	171
9	Extending Fibrations From Graphs to Hypergraphs	173
9.1	Introduction	173
9.2	Hypergraphs	173
9.3	Graph, hypergraph, and admissible graph representations of ODEs	176
9.3.1	Graph representation	177
9.3.2	Hypergraph representation	178
9.3.3	Admissible graph representation	179
9.3.4	Which representation do we use?	180
9.4	Weighted networks	181
9.4.1	Balanced coloring for a weighted network	182
9.5	Multiplex and multilayer graphs	183

9.5.1	Balanced coloring for a multiplex/multilayer weighted network	184
9.6	Using typing to restrict the class of homomorphisms	185
10 Fiber Bundles for Physics — Fibrations for Biology		191
10.1	Fiber bundles	191
10.1.1	Cartesian products and trivial fiber bundles	192
10.1.2	Principal bundles in physics	193
10.1.3	Nontrivial fiber bundles	194
10.2	Fibrations in algebraic topology and category theory	196
10.3	Graph products	198
10.4	Cartesian graph bundles	199
10.5	Graph fibrations—history	200
11 Symmetry and Robustness in Biological Networks		202
11.1	Homomorphisms for biological networks	202
11.2	Duplication events break a group symmetry but not a fibration	203
11.2.1	Speciation events break a group symmetry but not a fibration	204
11.2.2	Vulnerability of automorphisms under mutations	204
11.3	Understanding ‘structure \rightsquigarrow phenotype’	206
11.3.1	Segregation versus integration of function	206
11.4	Axiomatization of biology	207
11.4.1	Speculations on axioms for biology and the Erlangen Program	208
12 Geometrization of Biology		211
12.1	Curvature and connection in geometry	211
12.2	Curvature and connection in physics	213
12.3	Curvature and connection in a financial fiber bundle	214
12.3.1	From finance to physics	217
12.4	Curvature and connection in the electromagnetic fiber bundle	219
12.5	Curvature and connection in gravity	221
12.6	Curvature and connection in biology	221
12.6.1	Flow of information in a genetic network: the genetic ‘spiritus’	222
12.6.2	Curvature and connection in a genetic network	224
12.7	Curvature and connection in the <i>E. coli</i> genetic network	229
13 Software and Algorithms to Perform Fibration Analysis		233
13.1	Finding symmetries	233
13.2	Finding the automorphisms of a graph	234
13.3	Finding the fibration symmetries of a graph	236
13.3.1	Coarsest equitable partition and tests for graph isomorphism	237
13.3.2	The color refinement algorithm	239
13.3.3	The Cardon–Crochemore algorithm and canonical colorings	240
13.4	Algorithms to find all equitable partitions	242
13.5	How to extend fibration algorithms to take noise into consideration	246

13.5.1 Pseudosymmetries	246
13.5.2 Quasifibrations	249
13.5.3 Pseudo-balanced colorings and repair algorithm without knowledge of balanced coloring	251
13.6 How to extend fibration algorithms to take types into consideration	252
13.7 Software used in this book to find minimal balanced colorings	254
Part II Applications: from Genes to the Brain	261
14 Fibration Analysis of Biological Networks	263
14.1 Geometric biology	263
14.2 Transcriptional regulatory networks	263
14.3 Fibration analysis of the TRN	266
14.4 Searching for biologically meaningful building blocks	271
14.5 Definition of fibration building blocks	275
14.6 Classification of simple building blocks	281
15 Simple Fibration Building Blocks	285
15.1 The primordial building block: autoregulation (AR) loop	285
15.2 Dynamic repertoires in classes of fibration building blocks	287
15.3 The feed-forward fiber - FFF	289
15.3.1 Feed-forward loop (FFL) network motif	290
15.3.2 Feed-forward fiber synchronization	294
15.3.3 Satisfied feed-forward fiber (SAT-FFF) synchronization	295
15.3.4 Unsatisfied feed-forward fiber oscillates and synchronizes	298
15.3.5 UNSAT-FFF clock functionality	302
15.4 Simple multilayer fiber	305
15.5 Simple Fibonacci fibers	308
15.6 Binary tree fiber ($n = 2$) and n -ary tree fibers	311
15.7 Evolutionary dynamics of fibration building blocks	311
15.7.1 Complexity of the building blocks: fractal dimensions	313
15.8 Biologically trivial fibers: operons and regulons	314
15.9 Fibration building block landscape across networks and species	316
16 Fibration Analysis of Metabolic Networks: Complex Building Blocks	318
16.1 Metabolic networks	318
16.2 Enzyme network of <i>E. coli</i>	319
16.3 Fibration analysis of the <i>E. coli</i> enzyme network	319
16.4 Complex fibration building blocks	321
16.5 Composite Fibonacci fibration building blocks	323
16.5.1 Composite Feed-Forward Fibonacci building blocks	326
16.6 Fibration landscape of enzyme networks	328

17 Comparison Between Fibration Building Blocks, Motifs and Modules	330
17.1 Biological significance of fibers	330
17.2 Network motifs	331
17.2.1 Calculating network motifs	333
17.2.2 Evaluating <i>p</i> -values and Z-scores	334
17.2.3 Statistical significance of network motifs	336
17.2.4 Network motifs in complex networks	337
17.2.5 Drawbacks of network motifs	338
17.3 Statistical significance of fibration building blocks	341
17.4 Modularity	342
17.5 Comparison of fibrations, motifs, and modules in enzyme networks	346
17.5.1 Fibers and modules in TRNs	346
17.5.2 Fibers, modules and motifs in metabolic networks	348
18 Synthetic Biology Through Symmetries: the Cell as a Computer	352
18.1 Synthetic biology design through symmetry and broken symmetry	352
18.2 A biological transistor at the core of genetic circuits	354
18.2.1 Repressor interaction maps to a transistor	355
18.2.2 AR loop maps to a ring oscillator	355
18.2.3 FFF maps to a Widlar current-mirror electronic circuit	357
18.3 Gene duplication in evolution mimics lifting in a fibration	357
18.4 Symmetry breaking in physics	361
18.5 Hierarchy of broken symmetry circuits: duplication and memory	362
18.6 Lifting the AR base: toggle-switch or SR flip-flop circuit	363
18.7 Lifting the FFF base: clocked SR flip-flop circuit	364
18.8 Lifting the Fibonacci base: JK flip-flop circuit	365
18.9 The TRN as a computer	366
18.10 Algorithm to find fibration symmetry breaking circuits	367
18.10.1 Software to find broken symmetry circuits	367
19 Taming Biological Complexity with Symmetries: the Minimal Genome	372
19.1 Fibration complexity reduction	372
19.2 CoReSym: reducing the TRN to its minimal computational core	373
19.3 Symmetry fibrations reduce the network yet preserve information flow	378
19.4 Finding the driving core	378
19.4.1 Injective fibrations	378
19.4.2 <i>k</i> -core decomposition	379
19.5 Structure and composition of the minimal TRN: a simple computer	380
19.5.1 The minimal TRN	381
19.5.2 Toggle-switches (flip-flops) in the TRN	382
19.5.3 Oscillators (clocks) in the TRN	383
19.5.4 Additional electronic circuits: toggle-switch and clock-type	383
19.5.5 Structure of cycles	384
19.6 Assembling the TRN from its building blocks	386

20 From Structure to Function: Cluster Synchronization in Genetic Networks	395
20.1 Structure \rightsquigarrow function in genetic networks	395
20.2 Structural network and functional network	398
20.3 Cluster synchronization from gene coexpression	399
20.3.1 Thresholding method to find cluster synchronization	400
20.3.2 Hierarchical clustering to find cluster synchronization	405
20.4 Gene coexpression synchronization analysis from massive datasets	405
20.4.1 Gene synchrony in fibration circuits	406
20.4.2 Structure \rightsquigarrow function in the carbon utilization circuit of <i>E. coli</i>	408
20.4.3 Synchronization in the hierarchy of fibers in <i>E.coli</i> and <i>B.Subtilis</i>	410
21 All Biology Conspires Against Synchrony, so How Can We Be Alive?	413
21.1 Modeling in Biology	413
21.2 Synchronization in biology is not ideal	414
21.3 An ideal ODE model of gene expression dynamics	414
21.3.1 A more realistic ODE model of the FFF	415
21.4 The non-idealized ODE model does not lead to synchronization	416
21.5 The uniformity assumption: idealized FFF model leads to synchronization	417
21.6 Breaking of uniformity	419
21.6.1 How physicists deal with this problem	421
21.7 Is symmetry necessary in biological networks?	423
21.7.1 Weak symmetry breaking by differing gene input functions	424
21.7.2 Similarities between measured gene input functions	424
21.7.3 TF binding sites	428
21.8 Selection pressures for symmetry	429
21.9 Towards a gene regulatory code	431
22 From Function to Structure: Network Inference Reconstruction	432
22.1 Biological networks are never complete	432
22.2 Link prediction in biological networks	434
22.2.1 Previous reconstruction algorithms for missing data	437
22.2.2 Missing link algorithms using topological measures of the network	437
22.3 Inferring the pathways guided by cluster synchronization	439
22.4 Link prediction in biological networks via symmetry and synchronization	440
22.5 Integer linear program to infer the structural network from cluster synchronization	441
23 Fibration Theory of the Brain I: <i>C. elegans</i> Locomotion	445
23.1 Can we hope for an organizing theory of the brain?	445
23.2 Dynamics and structure \rightsquigarrow function in the brain	448
23.3 The <i>C. elegans</i> neural system	450
23.4 Synchronization in <i>C. elegans</i> neural system	451
23.5 Synchrony and symmetry in the worm	455
23.5.1 Whole nervous system recording and motor neuron characterization	456

23.5.2 How to measure synchronization	457
23.5.3 Measures of synchrony	458
23.5.4 Synchrony and correlation matrix	460
23.5.5 Clique synchronization	460
23.5.6 Modularity and community detection	461
23.5.7 Consensus cluster synchronization	461
23.6 Symmetry-driven reconstruction of the <i>C. elegans</i> locomotion connectome	461
23.7 Perturbations and symmetry breaking by ablation and inhibition	463
23.7.1 Future work	464
24 Fibration Theory of the Brain II: the Minimal Engram in Mice	465
24.1 One-to-many relation between structure and function in the engram	465
24.2 Where in the connectome are memories stored?	466
24.3 How to measure engram synchronization	467
24.4 Inferring the engram network	471
24.5 The minimal engram network	472
25 Fibration Theory of the Brain III: the Human Language Network	475
25.1 Structure and function in the human brain	475
25.2 Human language network	477
25.3 Synchrony in the human brain at the mesoscale scale	478
25.4 Patterns of synchrony during resting state and language task	480
25.5 From synchronization to the structural network	481
25.6 Symmetry breaking in the human language connectome	483
25.7 Application of symmetry theory to find essential areas in the brain	484
26 Fibrations in Artificial Intelligence	487
26.1 Geometric Deep Learning and the Erlangen Program	487
26.2 Symmetries in deep neural networks	488
26.3 Upper bound on the expressive power of GNNs	489
26.4 Fibration symmetries in graph-structured datasets	492
26.4.1 Training GNNs with minimal fibration base	493
26.5 Fibration symmetries and synchronization in training deep neural networks	494
26.6 Application of fibration symmetries to gradient descent	496
26.7 Fibration, opfibrations, and coverings in DNNs	498
27 Overview and Outlook	500
27.1 Overview	500
27.2 Outlook	501
Appendix A Software, code, and data	505
<i>References</i>	509
<i>Index</i>	535

PART I

THEORY: SYMMETRY GROUPS IN PHYSICS; SYMMETRY FIBRATIONS IN BIOLOGY

Part I introduces the concepts of symmetry fibration, balanced coloring, and cluster synchronization in networks, comparing these ideas to alternative frameworks based on symmetry groups. We demonstrate that the global group symmetries typically used to describe physical systems are too restrictive to fully represent the complexity of biological networks. In contrast, a more flexible local symmetry—specifically fibration symmetry—can effectively capture the structures within biological networks that lead to cluster synchronization. In Part II, we explore the applications of fibration symmetry to various biological networks, including genetic and metabolic networks as well as neural networks in the brain.

This chapter presents the central question we address in this book: the role of symmetry in our understanding of nature. Symmetry is fundamental in physics at all levels, from the building blocks of matter and the shapes of snowflakes to the fabric of spacetime. The effectiveness of symmetries in explaining the physical world leads us to question why these same concepts cannot account for emergent properties in biological systems. Given that the same physical laws apply to both living and non-living systems, we must ask: if life is an emergent property of physics, why cannot the symmetry principles of physics also explain the organization of life? We propose a more general notion of symmetry, fibration symmetry, that is better suited for this purpose, and explain how it relates to the important biological function of synchrony.

1.1 Symmetry in physics and biology

The mathematical scaffold of modern physics is built around principles of symmetry (Gell-Mann, 1995; Weinberg, 1995; Wilczek, 2016). Since nature seems to favor symmetry at a very fundamental level, symmetry has become a powerful tool for discovering new laws of nature. Physicists traditionally think of the notion of symmetry as being synonymous with the theory of groups and their actions, a beautiful and extremely powerful branch of mathematics. The theory of symmetry groups is the cornerstone of modern theoretical physics, on which our understanding of particles, forces, and matter has been built, via Lorentz invariance, Einstein's equivalence principle, gauge invariance, Noether's theorem, Bloch's theorem, Landau's theory of phase transitions, and so on (Weinberg, 1995; Landau and Lifshitz, 1977; Georgi, 2018; Dixon and Mortimer, 1996; Stewart et al., 2003; Golubitsky et al., 2005a).

In particular, symmetries and group theory lie at the core of the theoretical formulation of the Standard Model, which provides a unified description of all fundamental particles and their interactions (with the exception of gravity). This description is formalized by continuous (Lie) symmetry groups of the Lagrangian (Weinberg, 1995).

In crystallography, point symmetry groups systematically classify crystal structures (Vainshtein, 1994). Molecular symmetry groups in chemistry classify the three-dimensional arrangements of atoms and their chemical bonds that constitute all molecules (Landau and Lifshitz, 1977). Furthermore, the unifying mechanism of spontaneous symmetry breaking explains pattern formation in numerous areas of science, the existence of phase transitions

(e.g., ferromagnetism, superconductivity, and superfluidity), and the emergence of universal critical behavior unifying condensed matter and high-energy physics (Weinberg, 1995).

Symmetries are significant in theoretical physics, and this encourages exploring whether they can help us organize the vast amounts of information present in biological systems. These systems consist of interacting components arranged in networks, such as the billions of neurons in the human brain or the tens of thousands of genes, proteins, metabolites, and other biomolecules that form genetic, protein, metabolic, and signaling networks within an organism.

A key question is whether the structure of these biological networks has developed according to the same symmetry principles that govern molecules and matter. Furthermore, we need to investigate whether these symmetries can assist in identifying the functional building blocks of living networks, much like they help identify the fundamental building blocks in particle physics. If this proves to be the case, it could lead to a systematic organization of biological complexity, with significant implications for theoretical biology, similar to the role of symmetries in theoretical physics.

However, this program has not yet been done, strongly suggesting that the group-theoretic notion of symmetry so prevalent in physics is not an appropriate fundamental framework for biology. We aim to bring the concept of symmetry into the age of genomics and systems biology by exploring whether a more general notion of symmetry can describe the structure of biological graphs, including transcriptomes, proteomes, metabolomes, and connectomes. The new theory of fibration symmetries presented in this book extends the concept of global symmetry groups, transitioning into more flexible and local symmetries. It demonstrates that the fibration formalism serves as the ideal mathematical framework for describing biological networks.

1.2 Group-theoretic symmetries in biology

We are not suggesting that the ‘global’ notion of symmetry formalized in group theory is irrelevant to biology. A notable early example is the influential 1917 book *On Growth and Form* by D’Arcy Thompson (Thompson, 1942). In this all-time favorite, Thompson meticulously documented the symmetries found in the structural patterns and forms of simple living systems. He explored various examples, ranging from the shapes of viruses and the icosahedral structure of radiolaria to the left-right symmetry of bilaterians and the spirals of mollusks, as well as the Fibonacci phyllotaxis observed in sunflowers. Figure 1.1 shows one of Ernst Haeckel’s famous drawings of radiolaria from the *Challenger* expedition. Modern research has shown that Haeckel often idealized the symmetry (Jungck et al., 2019), but the figure is a dramatic illustration of the spirit of Thompson’s ideas. The body of literature surrounding Thompson’s concepts is huge and diverse; (Chaplain et al., 1999) provides a balanced assessment and modern update.

Thompson’s book was published prior to the discovery of DNA structure and the subsequent big-data ‘omics’ era in systems biology. While Thompson enumerates these beautiful geometric invariants, he does not offer insights into the molecular origins of such symme-

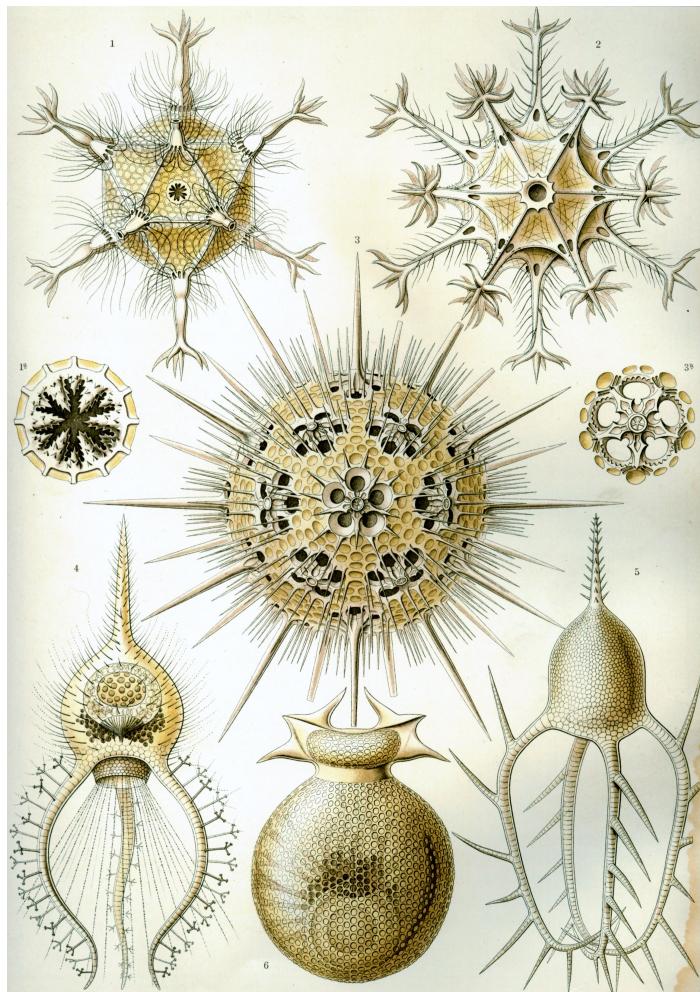


Fig. 1.1

The group symmetries of growth and forms in living systems are fascinating, yet they do not describe biological graphs. In his influential book, Thompson (1942) examines numerous examples of what we now call group symmetries within simple biological systems, particularly highlighted by the polyhedral forms of radiolaria that Ernst Haeckel illustrated in 1904, shown here. Unfortunately, these group symmetries cannot be translated to explain genetic graphs underlying life. Reprinted from Wikipedia <https://en.wikipedia.org/wiki/Radiolaria>.

tries at the level of networks involving genes, proteins, and other biomolecules, which are of interest here. Not surprisingly, given the era in which he worked, he focuses on symmetries of the *organism*, not on the underlying molecular biological *graphs*. Additionally, he overlooks the principles of Darwinian evolution, being somewhat antipathetic to it, on the grounds that it ignores physical constraints on form.

Thompson's book inspired Alan Turing's groundbreaking work on mathematical biology (Turing, 1952). Turing aimed to explain how symmetric pattern formation in living organisms emerges from a uniform state through the process of morphogenesis in catalytic chemical reactions. These patterns, known as 'Turing patterns' include examples such as the stripes of a zebra and the spirals in the skin pigmentation of a giant pufferfish. Turing's theory fell out of favor with the huge advances in molecular biology, but more recently it has been revived with the identification of specific molecules as 'morphogens' and modern experimental techniques (Murray, 1989; Meinhardt, 1995; Kondo and Asai, 1995; Sick et al., 2006; Economou et al., 2012; Sheth et al., 2012).

The search for symmetries in biology re-emerged during the Eleventh Nobel Symposium, chaired by Jacques Monod in 1968, and was documented in its Proceedings (Monod, 1970). During the Symposium, scientists discussed how the functions of macromolecules may arise from the symmetric arrangements of specific biomolecules. They explored how a protein's function could be encoded in the three-dimensional (quasi-) symmetric shape of its quaternary structure or how the functionality of an enzyme is determined by the highly symmetric coupling that allows a metabolite to bind at its active site. Additionally, it was acknowledged that various types of symmetry are observed at every level of the physical structure of biomolecules. Examples range from quantum mechanical atomic orbitals and the shapes of individual molecules to genetic sequences, the configurations of amino acids, and the alpha helices and beta sheets in proteins. Moreover, structures such as pores and proton pumps in cell membranes often exhibit symmetry, typically belonging to either a cyclic or dihedral group.

On the other hand, Monod and others also noted that biomolecular symmetries seldom extend beyond the level of quaternary structure of proteins. They play a role in early biological development; for instance, the symmetries of the fertilized egg or the blastula break to create important structure in the growing organism—but genetic effects quickly come into play as well. Beyond this level, the group symmetries of physics go silent when explaining biological function or even the gross shape of an organism.

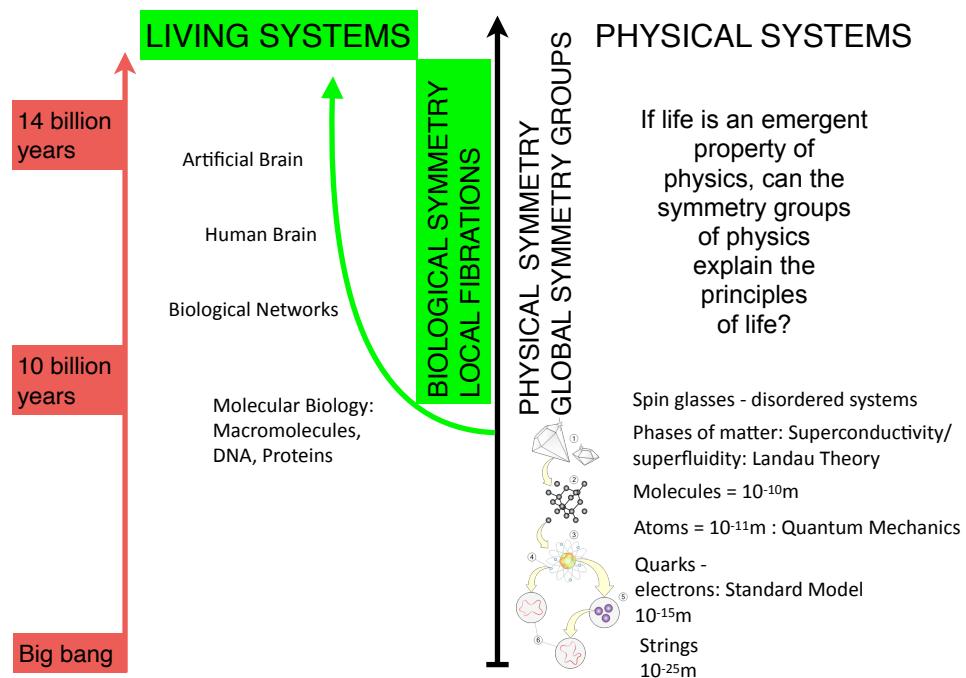
Two fundamental questions were raised at the Symposium and they remain unanswered (Monod, 1970; Atkinson, 1970):

Q1. *'Is there continuity in kind between the symmetries seen at the levels of the molecule, the organelles, the virus, the cell, the organism?' (Fig. 1.2).*

Q2. *'Are these symmetrical arrangements really useful only structurally, or might they be more directly related to dynamic function?'*

Despite raising the right questions at the symposium, the hypothesis that biological function could emerge from symmetry quickly fell into oblivion afterward. This may have happened because J. D. Bernal—another towering figure in molecular biology—gave an emphatic 'no' in response to **Q1**. He pointed out that 'the symmetry of a sea star does not arise from self-assembly of the arms' (Atkinson, 1970), and further stated (Bernal, 1967): '*There is radical difference between symmetry as observable in organisms and that in crystals*'.

More recent efforts have focused on applying group symmetries to dynamical systems within the field of systems biology. These studies encompass areas such as locomotion,

**Fig. 1.2**

Group symmetry in physics and fibration symmetry in biology. The leitmotiv of this book is to explore the question posed by Morone et al. (2020): ‘If life is an emergent property of physics—why cannot the same symmetry principles that explain physical phenomena also explain the organizing principle of life?’ The success of group symmetries in elucidating the physical world—from general relativity to the standard model of particle physics and all phases of matter—raises an important question about their applicability in understanding the emergent properties of biological systems. In this book, we argue that a relaxed form of local symmetry, called symmetry fibration, explains the structure and synchronization observed in biological networks.

evolution, and visual hallucinations (Golubitsky and Stewart, 2003, 2015). A survey examining symmetry in neuronal circuits from the perspective of ‘form and function’ can be found in (Stewart, 2022). We shall not provide a comprehensive review of such work here. However, these applications tend to be specialized, often rely on idealized models, and do not adequately tackle the large data sets currently available, like gene regulatory networks and connectomes. To date, these individual areas have not had the impact on biology that group-theoretic symmetry has had on physics.

1.3 A more general notion of symmetry

It therefore seems worthwhile to seek a more general concept of symmetry, better adapted to the needs of ‘big data’ biology. The present book aims at resuming the line of research started by Thompson, Turing and Monod, by bringing new symmetry principles, not based on groups, into the era of big-data ‘omics’. Bernal, like other historical figures of his time, did not have access to the vast amount of large-scale high-throughput data on all branches of functional genomics that is available to present-day systems biologists. We will show how the principle of network symmetry, formulated in an appropriately flexible and local manner, augments modern graph theory. This provides a theoretical method for identifying functional building blocks of biological synchronization and for understanding how the structure of a network determines cellular function.

This more general notion of symmetry does not displace or compete with the existing group-theoretic one: instead, it complements it. Its mathematical origins go back a century to Brandt (1927), who introduced the notion of a ‘groupoid’ and established many basic properties; it has been employed increasingly in pure mathematics ever since, especially in algebraic topology (Brown, 2006). Its applications in science are more recent; in the biological context just described, it turns out to be a natural way to approach key biological issues of form and function.

Traditionally, the symmetries of a network have been formalized as permutation symmetries of the nodes of the graph that leave its adjacency matrix invariant. In simple terms, these symmetries are those permutations of nodes that preserve the global connectivity of the network. Each of these permutation symmetries imposes strong conditions on the dynamics of the nodes in the graph because symmetry permutations leave invariant any ‘admissible’ equation—one that is compatible with the network topology—that describes the dynamics of the system. Nodes that are permuted by the symmetries are divided into the ‘orbits’ of the symmetry group acting on the graph, and these orbits form clusters of synchronized nodes. This phenomenon is known as ‘cluster synchronization’.

These permutation symmetries are called automorphisms. They form a group called the symmetry group of the graph: each symmetry has an inverse, their composition is associative, and the identity function is a symmetry permutation. Symmetry groups are the theoretical basis of all symmetries in physics, yet they have found comparatively few applications in biology. At the same time, cluster synchronization is ubiquitous in biological networks since the collective dynamics of synchronized units has biological significance for their function.

Box 1.1**The conundrum of this book**

How can we rationalize the widespread existence of cluster synchronization found in all biological systems in the absence of symmetry groups in their underlying biological networks?

The main contention of this book is that there is a more general form of network symmetry known as ‘fibration symmetry’ (Morone et al., 2020; Leifer et al., 2020). This new type of symmetry is local, allowing for a broader range of synchronous clusters compared to global group symmetries. Mathematically, fibration symmetry is both necessary and sufficient for robust cluster synchrony (Golubitsky and Stewart, 2023). Here ‘robust’ means that such clusters occur for any admissible ODE. However, *existence does not guarantee stability*, which depends on the specific model ODE under consideration. On the other hand, any steady-state pattern of synchronous clusters that arises from a fibration is stable for *some* admissible ODE.

Fibration symmetry, therefore, explains the widespread existence of cluster synchronization in biology in the absence of traditional symmetry groups. (We use both terms ‘symmetry fibration’ and ‘fibration symmetry’. They are essentially synonymous, but the former focuses on the type of fibration, while the latter focuses on the type of symmetry.)

Box 1.2
Our working hypothesis

The function of biological systems is engraved in the fibration symmetries of the underlying biological networks that make up the genome, transcriptome, proteome, metabolome, and the brain connectome, which define life at the system level.

Symmetry fibrations offer a more general and less restrictive form of symmetry than automorphisms. They are not associated with symmetry groups but with symmetry groupoids and are formalized by graph fibrations (Grothendieck, 1959; Boldi and Vigna, 2002a; Golubitsky and Stewart, 2006, 2023). Graph fibrations are homomorphisms—structure-preserving maps—between graphs, subject to some simple but vital extra conditions. These transformations do not preserve the adjacency matrix of the network, as automorphisms do, but they define a natural way to ‘collapse’ sets of nodes into clusters. Automorphisms are homomorphisms but with extra restrictions. Graph fibrations can be of many types: injective, surjective, bijective, or none of those, and minimal or not minimal. Not all of these possibilities are relevant to biological systems. Only one particular type of graph fibration has a profound biological meaning because it can reduce the network to its fundamental building blocks for cluster synchronization, as shown in Fig. 1.3. These graph fibrations are surjective and minimal, and they were named by Morone et al. (2020) as *symmetry fibrations*. They are closely related to balanced colorings and initially we use this term informally through examples, without defining it.

1.4 Cluster synchronization and fibrations

The previous discussion implies that the existence of synchronized clusters of nodes in biological networks is not accidental but follows from the presence of fibration symmetries

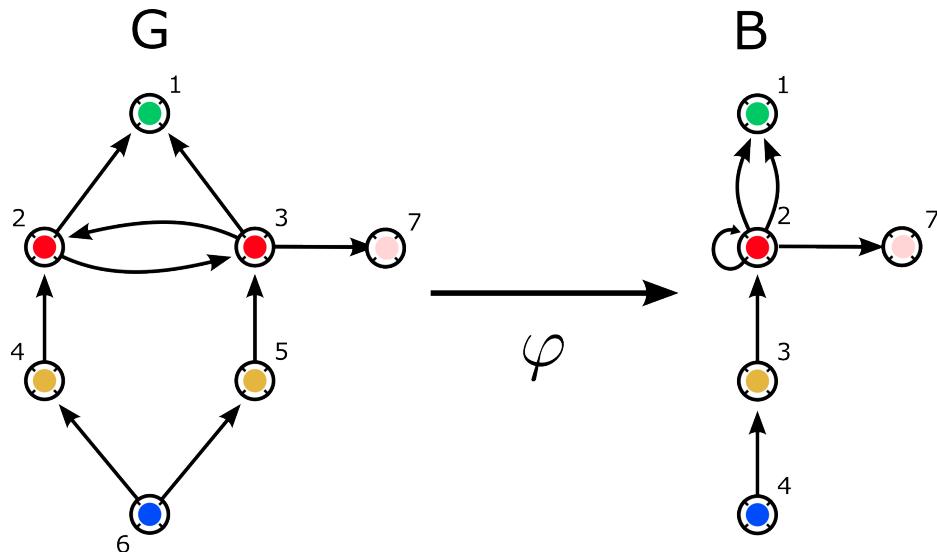
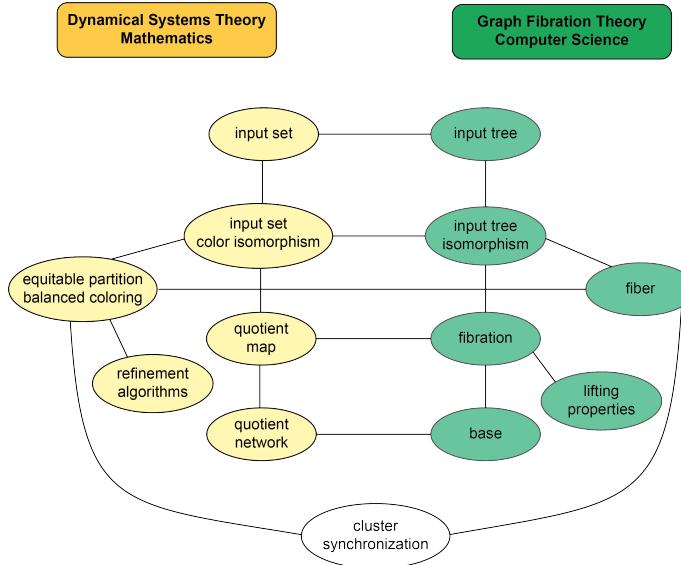


Fig. 1.3

A symmetry fibration φ from a graph G to its base B . The graph G can represent any biological network, such as a set of regulating genes or interacting neurons. It has no automorphism (no group symmetry) except for the identity. Yet, it possesses a rich fibration symmetry. The colored nodes demonstrate a balanced coloring of the graph. Equal-colored nodes receive the same 'amount' of colors from their in-neighbors, ensuring cluster synchronization among nodes of the same colors in the absence of a symmetry group. The fibration symmetry φ collapses these clusters (called fibers) while preserving the dynamics of graph G into the base B . Edges $(2, 3)$ and $(3, 2)$ are collapsed into the loop $(2, 2)$ and two edges $(2, 1)$ and $(3, 1)$ become a multi-edge $(2, 1)$. This book elaborates on these concepts and their application to biological networks.

in the network, which apply to any compatible dynamical system. These clusters play a fundamental role in understanding living systems because synchronization is an essential component of biological processes: cluster synchronization of biological units (proteins, neurons, etc.) indicates their participation in a common function. Cluster synchronization appears in synchronized gene expression patterns (Eisen et al., 1998; Langfelder and Horvath, 2008) and the synchronized oscillatory dynamics of ensembles of neurons in the brain (Singer, 1999; Uhlhaas et al., 2009). Synchronization and oscillations further appear in circadian rhythms, the cell-division cycle, the sleep-wake cycle, respiration, locomotion, and cardiac rhythms, to name just a few (Strogatz, 2018; Golubitsky and Stewart, 2006).

Graph fibrations are analogous to, and inspired by, topological fibrations, which were introduced by mystic genius Grothendieck (1959) within category theory. The graph-theoretic analog was developed by Boldi and Vigna (2002a) in the context of distributed systems. In successively more general form, Stewart et al. (2003); Golubitsky et al. (2005a) and Golubitsky and Stewart (2023) developed an alternative formalism to fibrations based on balanced

**Fig. 1.4**

Main mathematical concepts. In Part I of this book, we examine key mathematical concepts as they are understood by various communities involved in the development of these theories. Mathematicians, computer scientists, and dynamical system theorists often use different terminology for the same concepts which are all related as shown here. We aim to unify these diverse groups within a common framework as shown in the graph, which is further developed to enhance the understanding of biological systems.

colorings of a graph and its symmetry groupoids. Balanced colorings of graphs, also called ‘equitable partitions’, are an alternative, mathematically equivalent way to describe fibers in the fibration formalism (Fig. 1.3). They therefore lead to the same robust cluster synchrony patterns. Thus, both fibers and balanced colorings describe, in two different ways, the same synchronous dynamics of clusters of nodes.

Fibrations and balanced colorings were shown to be equivalent by Aldis (2008). The equivalence between fibration symmetry and groupoid symmetry was demonstrated by Rink and Sanders (2013); DeVille and Lerman (2015b) and discussed in (DeVille and Lerman, 2015a; Nijholt et al., 2016). An approach from an engineering viewpoint was investigated by (Field, 2004) and developed in (Agarwal and Field, 2010b,a). The connection between these approaches and the nomenclature relating the formalisms of fibrations, groupoids, and balanced colorings are shown in Fig. 1.4, and discussed in detail in the rest of the book.

Figure 1.3 shows an example of a graph G with no automorphisms (except for the identity), yet with a symmetry fibration that determines fibers, which are the balanced coloring shown by the differently colored nodes. Given any admissible system of dynamical

equations associated with the graph, it becomes possible for these fibers to synchronize their activity. This synchrony pattern cannot be captured by automorphisms. The symmetry fibration collapses the graph G onto its base B as shown. In succeeding chapters we discuss such fibrations extensively, and characterize admissible dynamics.

In this book we show that symmetry fibrations define the key features of the structure of biological networks. We show that many biological networks have fibration symmetries and display dynamical invariances, despite the absence of automorphisms. We further show that the requirement of integration of functionally segregated modular units in biological systems strongly suppresses the applicability of automorphisms to biology.

Applications of networks are not restricted to biology but also include social networks, financial networks, the Internet, infrastructure networks, and ecosystems, among others. Studying each of these networks separately is a fascinating problem, but having a unified approach to study the dynamics of all such networks from a symmetry point of view, without needing to know how to describe the dynamics of each node, is very beneficial. Groupoids and fibrations describe the symmetries of any network, where each node of the network defines an interacting unit, and an edge between the nodes represents the transmission of information—a signal of physical contact from one node to the other, realized as the influence of one node on the dynamics of another.

Morone et al. (2020) and Leifer et al. (2020) have applied fibration symmetries to information-processing networks to discover functional building blocks of biological networks and to study their functions. This book describes these discoveries in a coherent fashion, as well as the theoretical machinery behind the fibration approach that has been developed over the last few decades in the fields of dynamical systems. This is done in Part I in a simple yet rigorous way to make the concepts and methods available to a wider audience in the fields of graph theory, complex networks, and biology, as well as to connect the dots between those fields for more mathematically inclined readers. Part II of this book deals with concrete applications to biological systems ranging from genetics networks to the brain.

2

Symmetry Fibrations in a Nutshell

If all you have is fifteen minutes and want to grasp the essence of fibrations, this chapter is for you. We explain, in simple terms, how fibrations generalize the concept of automorphisms—that is, symmetry groups—and why it is difficult to realize these symmetry groups in biological networks. The central point of this book is that fibration symmetry, a local and less rigid form of symmetry, represents the structure of biological networks better than global symmetry groups do. In particular, it controls the important feature of cluster synchronization. We introduce the minimal concepts needed to understand fibrations and their relation to cluster synchronization, concepts that are elaborated later in the book.

2.1 Informal introduction to fibrations

In this chapter, we introduce, in an informal manner, a small number of key mathematical concepts that determine and describe fibrations and cluster synchronization in networks. We do this in the context of biological networks, specifically simple gene regulatory networks (GRNs), because the primary target audience of the book is the biological community. A secondary aim is to introduce the biological and mathematical concepts to mathematicians, physicists, computer scientists, and engineers, who are interested in applications to biology (or, indeed, to other areas of ‘real world’ science).

Because of this diverse audience, we avoid technical definitions in this chapter. Instead, we use simple but typical examples to illustrate the main ideas, which are treated in more detail in subsequent chapters. The mathematical concepts can be defined, and analyzed, in much greater generality, but we leave these extensions for later. The ideas presented here are repeated later in Part I in greater depth and through a richer set of examples. Part II focuses on applications of fibrations in biology and artificial intelligence.

Research into biological networks has focused on the importance of synchronization (also referred to as synchrony). Nodes in a network are *synchronized* if they display identical dynamics. A significant amount of research in dynamical systems has focused on understanding *complete* synchronization in networks. Complete synchronization occurs when all nodes in the network are synchronized in the same state. This research primarily examines dynamics described by the Laplacian matrix: $L = D - A$, where D is the degree matrix, and A is the adjacency matrix of the graph (Pecora and Carroll, 1990). Laplacian dynamics describe networked systems interacting with diffusive couplings (like the Kuramoto model, see Section 3.6.1).

However, complete synchronization is rarely observed in biological networks. Cluster

synchronization, which occurs when nodes split into disjoint subsets, such that they synchronize within each subset but do not synchronize between different subsets, is mathematically more common and also biologically more significant.

Examples are abundant in biology. In a GRN, the genes concerned are active in clusters at the same time and in the same manner, and this cluster synchronization constitutes a basic type of functionality (Leifer et al., 2020). Hebbian learning (Hebb, 1949) is based on the principle ‘neurons that fire together, wire together’, and the neurons concerned fire (or do not fire) at the same time in neuronal assemblies. In any network, cluster synchronization reveals functional building blocks of the network structure (Morone and Makse, 2019; Morone et al., 2020; Leifer et al., 2020).

These biological systems are not described by the diffuse couplings of Laplacian dynamics. Instead, the dynamics in the biological systems treated in this book are defined in terms of the adjacency matrix of the network and their input trees, whose symmetries explain the cluster synchronization.

The area of network dynamics is becoming increasingly important for several reasons. Scientists (and indeed other researchers) have come to realize that many real-world systems are naturally structured as networks: GRNs, neuronal networks in the brain or elsewhere, predator-prey networks, evolutionary trees, epidemic networks, the Internet, social networks, transportation networks, economic networks... All of these networks benefit from the understanding of symmetry and synchrony.

2.2 Fibrations, input trees, balanced colorings, and synchrony

The notions discussed in this section can be formulated in three distinct but equivalent ways: as (a) *synchrony patterns*, (b) *fibrations, fibers* and *input trees*, and (c) *balanced colorings* with their associated *quotient networks* or *bases*. All three formulations occur in the literature, and it is important to understand how they are related.

Synchrony between nodes is defined as nodes that exhibit identical time-series and synchrony clusters can be obtained using *graph fibrations*. The synchrony clusters form a partition of the network that is balanced colored (nodes with the same colors receive the same colors from their neighbors, Figs. 1.3 and 2.1) and nodes in a synchrony cluster (also called fiber) have isomorphic input trees (Fig. 2.2).

If G and B are graphs, a fibration $\varphi : G \rightarrow B$ is a map that sends nodes to nodes, arrows to arrows, and preserves input sets (where the input set of a node consists of all of its input arrows). Since fibrations preserve input sets, they also preserve input trees. Figure 2.2 is a simple example. Nodes 2 and 3 belong to the same fiber (pink), and they have isomorphic input trees, shown there to level 3, but the isomorphism is valid to all levels.

We elaborate on these definitions next.

2.3 Symmetry fibration in *Escherichia coli*

We begin by analyzing a simple example of synchronization in a tiny part of the genome of the bacterium *Escherichia coli*, one of the standard model organisms in genomics. Figure 2.1a illustrates a typical simple example of this situation: it shows a small subnetwork of the gene regulatory network of *E. coli*. The transcriptional regulators ExuR, UxuR, and LgoR control d-galacturonate and d-glucuronate metabolism (Tutukina et al., 2016; Shimada et al., 2018).

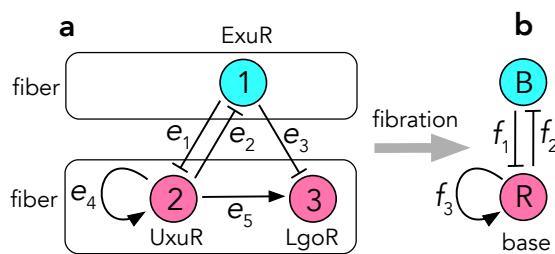


Fig. 2.1

Sample network to define the main concepts. (a) Subnetwork of GRN of *E. coli*, extracted from the *E. coli* TRN (Transcription Regulatory Network) controlling the gene *Uxur*. Colors show a synchronization pattern. (b) The corresponding base in which nodes of the same color are ‘collapsed’ to a single node. The gray arrow indicates a fibration: a map from the GRN to its base that preserves the directed edges that input to any given node. Curved rectangles outline the two fibers. Sharp arrow = activator, barred arrow = repressor.

For simplicity we assume that the subnetwork concerned has no external inputs; that is, we consider only the dynamics of this network on its own. We also assume that nodes 1, 2 and 3 have the same internal dynamics: they react to similar stimuli in similar ways. Edges e_1, e_2, e_3 determine identical repressor couplings, and edges e_4, e_5 determine identical activator couplings.

This 3-node network has no group symmetry. More formally, there is only a trivial group symmetry: there is no automorphism (permutation) of the network that preserves its topology except for the identity.

However, there exists a synchrony pattern in which nodes 2 and 3 are synchronized (see Section 2.4). (Node 1 typically is not synchronized with any other node.) How can this cluster of synchrony arise in the absence of automorphisms? The reason is that, just as for a group symmetry in the usual sense, if nodes 2 and 3 are synchronized, then they both receive the same input signals, hence remaining synchronized for all future time. To verify this claim, we can ignore node 1 since it is not expected to synchronize with any other node. Node 2 receives inputs from edges e_1 and e_4 , whereas node 3 receives inputs from edges e_3 and e_5 . However, the signals along edges e_1 and e_3 are identical because they both emanate

from node 1 and are both repressors; the signals along edges e_4 and e_5 are identical because they both emanate from the same node and are both activators.

We now formalize this intuition and explain the details included in Fig. 2.1, which relate to a general explanation of this type of cluster synchronization.

In the example network of Fig. 2.1, edges have a specific type (in the example, the types are ‘repressor’ and ‘activator’). When there is more than one edge type involved, the types must be taken into consideration when looking at the symmetry.

The colors ‘red’ and ‘blue’ partition the nodes into subsets $\{1\}$ and $\{2, 3\}$, which we call ‘fibers’. Subfigure (b) shows a simpler network, the ‘base’. It has only two nodes B (blue) and R (red), one for each fiber (hence for each color), repressor edges f_1 and f_2 , and an activator edge f_3 .

The large gray arrow in Fig. 2.1 indicates that subfigures (a) and (b) are related by a ‘fibration’. This is a map that collapses fibers to single nodes while retaining the set of input edges. More precisely, it is a map φ from the GRN to its base, which, among other properties described immediately below, maps nodes to nodes and edges to edges, while preserving sources and targets and also preserving the types of edges (here, activator/repressor). It does so according to the following scheme for nodes and edges, where B = blue, R = red:

$$\begin{array}{lll} 1 \mapsto B & & 2, 3 \mapsto R \\ e_1, e_3 \mapsto f_1 & e_2 \mapsto f_2 & e_4, e_5 \mapsto f_3 \end{array}$$

In general, the ‘fiber’ of a node in the original network is the set of all nodes that map to a given node in the base. In Fig. 2.1 the fibers are $\{1\}$, the only node that maps to B , and $\{2, 3\}$, the two nodes that map to R . The fibers correspond to the colors and determine the clusters in the corresponding synchrony pattern, which exists provided the map satisfies two additional properties, as we now explain.

Any map of this kind is called a ‘graph homomorphism’. However, φ has three further properties that make it a fibration:

1. φ is ‘surjective’: every node in the base is the image of some node in the GRN.
2. φ preserves the structure of ‘input sets’, the set of all edges with a given target node.
3. φ preserves the colors.

The above conditions mean that (under general assumptions) fibrations preserve synchronous dynamics. That is, when nodes of the same color are synchronized, the dynamics in the GRN is the same as the dynamics in the base.

We verify these two conditions for this example. For condition 1:

$$\begin{aligned} \text{Node 1 has input set } & \{e_2\} \\ \text{Node B has input set } & \{f_2\} \end{aligned}$$

and these match each other, both in type (one repressor) and in the color of the source nodes (blue). Similarly

$$\begin{aligned} \text{Node 2 has input set } & \{e_1, e_4\} \\ \text{Node 3 has input set } & \{e_3, e_5\} \\ \text{Node R has input set } & \{f_1, f_3\} \end{aligned}$$

and these match each other, both in type (one repressor and one activator) and in the color of the source nodes (blue and red respectively).

These conditions guarantee that if nodes 2 and 3 are synchronized initially, then they receive synchronized signals, hence remain synchronized in the future.

Translated into the terminology of colorings, these statements show that:

Nodes of the same color have inputs that match each other, as regards both the type of edge and the color of the source of that edge. That is, nodes of the same colors ‘receive’ the same colors from in-neighbors.

Such a coloring is said to be a *balanced coloring*.

Mathematically, the notions of a fibration and a balanced coloring are equivalent, and the fibers correspond exactly to the colored clusters, that is, the sets of nodes that synchronize according to the pattern of colors.

2.3.1 Input trees

In the fibration formalism, the behavior of each node is encoded in its input tree, which is a rooted tree that contains all the paths in the network that terminate at the node (Morone et al., 2020). This encoding is more convenient than the typical graph representation of the adjacency matrix, because it allows for the separation of the whole network into parts so that the symmetries that directly preserve the dynamics can be analyzed separately.

Since fibrations preserve input sets, they also preserve input trees in the following sense: If $\varphi : G \rightarrow B$ is a fibration, then the input tree of any node i of G is isomorphic to the input tree of its image $\varphi(i) \in B$, where B is the base. If φ is not surjective, there could be nodes in B with input sets (hence also input trees) that differ from those in G . Figure 2.2 is a simple example. Nodes 2 and 3 belong to the same fiber (pink), and they have isomorphic input trees, shown in Fig. 2.2 to level 3 (the isomorphism is valid to all levels)

A *symmetry fibration* is a surjective fibration that transforms the graph into its *minimal* base by collapsing *all* the nodes with isomorphic input trees. That is, the symmetry fibration produces the minimal base (with minimal colors) by collecting all the symmetries of the graph. The collapsed nodes belong to a fiber of the graph and, through the application of the symmetry fibration, the fibers are collapsed into one representative node at the base of the graph.

For example, Fig. 2.2 shows the three input trees for the graph in Fig. 2.1. The input trees for nodes 2 and 3 have the same topology, while the input tree for node 1 has a different topology. Indeed, the colors on the input trees for nodes 2 and 3 correspond as well. Clearly, such an isomorphism of input trees is necessary for a fibration to exist. In fact, this condition is also sufficient: isomorphism of input trees determines the *minimal fibration*—the one with the fewest colors.

Invariance of the input tree under a symmetry fibration preserves the dynamics of the system. Intuitively, this happens because directed paths in the graph correspond to the flow of information through the network. Section 2.4 makes this idea precise. This invariance has important consequences for the dynamics of the nodes, since it predicts robust cluster

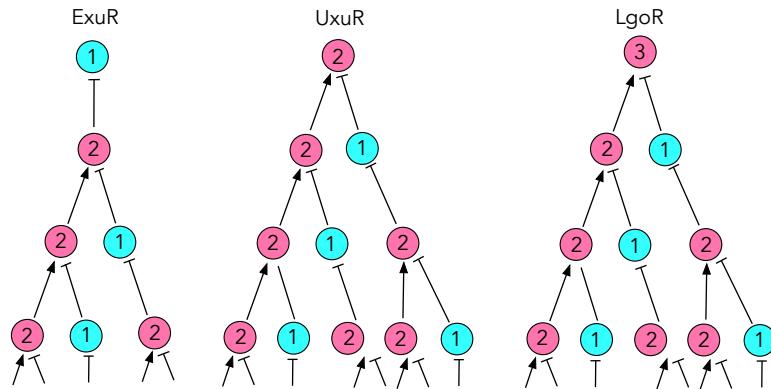


Fig. 2.2

Input tree examples. Input trees for the *E. coli* network of Fig. 2.1. Nodes 2 and 3 have isomorphic input trees, whereas that of node 1 is different. Thus, nodes 2 and 3 can synchronize, but node 1 cannot synchronize with either node 2 or node 3. Colors show the synchrony pattern but are not part of the definition of an input tree.

synchronization of nodes in the same fiber. Nodes in the same fiber synchronize their activity via the fibration.

A fibration symmetry is not a group symmetry of the graph: it is a symmetry that relates input trees. It represents invariances of the transmission of information through the network, which explains its widespread appearance in information processing networks in biology. In this book, we show that the synchronized fibers identified by symmetry fibrations are the fundamental building blocks from which biological networks are built.

2.4 Admissible equations and cluster synchronization

To illustrate the mathematics in a biological context, we consider a ‘toy’ model of the GRN of Fig. 2.1, employing Hill functions and linear interactions to represent gene dynamics. For simplicity, we assume a ‘lumped’ model in which the protein and mRNA concentrations are combined into a single real variable:

$$\begin{aligned}\dot{x}_1 &= -ax_1 + b \frac{1}{1+x_2^2} \\ \dot{x}_2 &= -cx_2 + b \frac{1}{1+x_1^2} + d \frac{x_2^2}{1+x_2^2} \\ \dot{x}_3 &= -cx_3 + b \frac{1}{1+x_1^2} + d \frac{x_2^2}{1+x_2^2}\end{aligned}\tag{2.1}$$

Here x_i is a measure of the activity of node i . The constants $a, b, c, d > 0$ are parameters whose values determine the amount of activation or repression and degradation rates. The minus signs on a, c indicate degradation. These terms represent the internal dynamics of

each node. The Hill function $\frac{1}{1+x^2}$ corresponds to repression, while $\frac{x^2}{1+x^2}$ corresponds to activation.

To build the ODE model of the graph Fig. 2.1, we have made several important hidden assumptions of the graph representation. For instance, we consider each edge to determine only the contribution of one input to the input function of the node and that these input edges are additive in the input function. Thus, individual arrows in the graph represent individual terms in the ODE. They can be set up that way for additive interactions. Also, the same parameters b, c, d appear several times because arrows of the same type model identical couplings. These assumptions and parameter choices will be discussed in more detail in Section 9.3 and Chapter 21.

To investigate synchronized solutions with the given coloring, we set $x_1(t) = x(t)$ and $x_2(t) = x_3(t) = y(t)$. Equations (2.1) become:

$$\begin{aligned}\dot{x} &= -ax + b \frac{1}{1+y^2} \\ \dot{y} &= -cy + b \frac{1}{1+x^2} + d \frac{y^2}{1+y^2} \\ \dot{y} &= -cy + b \frac{1}{1+x^2} + d \frac{y^2}{1+y^2}\end{aligned}\tag{2.2}$$

Such a system would be ‘overdetermined’ in general; that is, there would be more equations than unknowns. However, in this case, the repeated equation (for \dot{y}) is the same on both occasions, so no contradiction arises. Deleting the repeated equation for \dot{y} implies that any solution of the *restricted equation*

$$\begin{aligned}\dot{x} &= -ax + b \frac{1}{1+y^2} \\ \dot{y} &= -cy + b \frac{1}{1+y^2} + d \frac{y^2}{1+y^2}\end{aligned}\tag{2.3}$$

corresponds to a synchronized solution of (2.1) by setting

$$x_1(t) = x(t) \quad x_2(t) = x_3(t) = y(t)$$

and conversely. Moreover, (2.3) represents an analogous model for the base network.

Equations (2.2) are not the most general ones that are consistent with the network topology. They assume that interaction terms are additive, with specific forms, and that each arrow or node symbol determines one such term. The general theory of network dynamics shows that persistence of the synchrony pattern is not just an accident of this type of model. It holds for *any* differential (or difference or delay differential) equation model that respects the network topology and our basic modeling assumptions. Such a model is said to be *admissible* for the network topology.

The relation between (2.1) and (2.3) implies that the synchrony pattern described by that equation is valid for all times t . However, it is important to understand that this statement does not imply that the synchrony pattern concerned is dynamically *stable*—a property that is necessary for a given synchrony pattern to occur in reality. In practice the synchrony pattern persists, as time passes, only when small perturbations do not have a significant effect on the dynamics. If such perturbations preserve synchrony, or they break it but die down, then the synchrony pattern is stable. But if perturbations that break the synchrony tend to grow, the synchronized state is unstable.

Conditions for stability are not determined by the network topology alone but also by

details of whichever admissible ODE is used as a model. However, there are some general principles that characterize how the topology constrains stability. We return to this crucial point in Chapter 8.

2.5 Key principle

We now state a key principle for the existence of (not necessarily stable) synchrony patterns:

Synchronized nodes must have the same internal dynamics and must receive synchronized signals.

This principle is highly intuitive since differences in internal dynamics or incoming signals are likely to break the synchronization. This principle turns out (subject to mild technical conditions that exclude ‘accidental’ synchronies permitted by special kinds of coupling) to be equivalent to any of the following statements:

- Synchrony clusters determine a fibration.
- Synchrony clusters determine a balanced coloring.
- The subspace of all possible synchronized states is invariant under all admissible equations (i.e., those that respect the network architecture).

In a sense, everything else in this book follows from the above principle or expands on its implications and applications.

2.6 Global symmetry of automorphisms versus local symmetry of fibrations

We now discuss the two different notions of ‘symmetry’. The first is the global group-theoretic symmetry prevalent in physics. The second is the fibration symmetry which is local and usually more suitable for biology.

Global symmetries

Recall that in mathematics and physics, a ‘symmetry’ of some object normally means a transformation that preserves specified structural features of that object. A sphere has ‘spherical symmetry’, meaning that it looks the same if we rotate it about its center (or reflect it in a plane through its center); see Fig. 2.3a.

Similarly, a square has ‘square symmetry’, meaning that it looks the same if we rotate it about its center through one or more right angles, (or reflect it in a diagonal or a line bisecting opposite sides); see Fig. 2.4a.

The symmetries of any object, in this sense, form an algebraic structure, which, as previously mentioned, is called a ‘group’. One key property of groups is that if two symmetry transformations are composed—performed one after the other—the result is also a symmetry. This is the *closure property*. For example, if we rotate a sphere and rotate it again, the combined effect is the same as some other rotation. The same goes for a square.

Figures 2.3 and 2.4 involve *geometric* symmetries, and the symmetry transformations concerned are global motions in space. These global transformations preserve the global structure of the objects. But the concept of a global symmetry is more general. In particular, the corresponding concept for a graph/network is an *automorphism*. This is a *permutation* of nodes that preserves the global graph connectivity. It follows that any two automorphisms compose to give an automorphism—the closure property again—so the set of such permutations forms a group.

Figure 2.5 illustrates this idea using a graph with four nodes. In (a) we draw it as a square, which makes the analogy with a geometric square obvious. In (b) we draw a graph with exactly the same topology, which no longer looks like a geometric square. Because the two drawings have the same topology, they have the same automorphisms. For example, the geometric right-angle rotation corresponds to the permutation

$$\sigma = \begin{pmatrix} 1 & 2 & 3 & 4 \\ \downarrow & \downarrow & \downarrow & \downarrow \\ 2 & 3 & 4 & 1 \end{pmatrix}$$

For either drawing, this permutation ‘preserves’ connecting edges—that is, it preserves the graph topology—in the sense that node i is connected to node j if and only if $\sigma(i)$ is connected to $\sigma(j)$. For example, 1 is connected to 2, and $\sigma(1) = 2$ is connected to $\sigma(2) = 3$. It is easy to check that the four connections, and only those, are preserved.

Local symmetries

Figs. 2.3 and 2.4 show that even a very small change to a shape can destroy its symmetry. For example, if we cut a tiny hole in the sphere, as in Fig. 2.3b, it no longer looks the same after it is rotated. If we cut a tiny piece off a square, as in Fig. 2.4b, it no longer looks the same after it is rotated. These examples show that a symmetry, in this sense, is a *global* property of the entire object. We have seen that in mathematics and physics, global symmetries are formalized using the notion of a *group*.

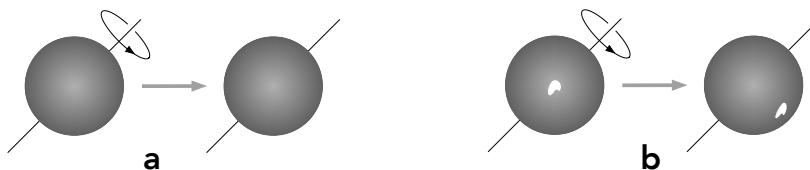


Fig. 2.3

Global symmetry of a sphere. (a) Rotational symmetry of a sphere. (b) One small hole in the sphere destroys the rotational symmetry.

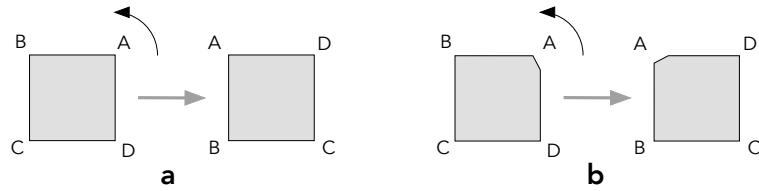


Fig. 2.4

Global symmetry of a square. (a) Rotational symmetry of a square though one right angle. (b) One small change to the square destroys the rotational symmetry.

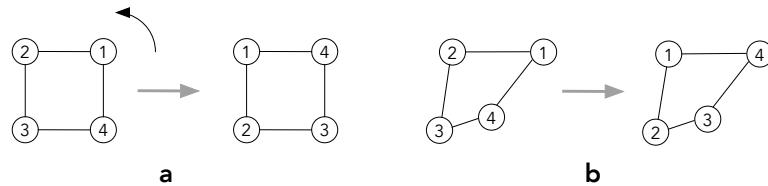


Fig. 2.5

Global symmetry in a graph. (a) This 4-node graph has the same symmetries as a square. (b) Any graph with the same topology also has the same symmetries as a square, despite being drawn as a different shape.

Although the group notion of symmetry is widely used in physics and mathematics, there are features that it fails to capture. For example, most regions of the sphere in Fig. 2.3b look the same. And three corners of the square in Fig. 2.4b are identical right angles. The rigid, global notion of a group symmetry does not include these resemblances. So there seem to be ‘local symmetries’, not captured by the usual notion of symmetry. (One might consider the concept of pseudo-symmetry in this case, as discussed in Section 13.5.1; however, this alone will go nowhere.)

A major point of this book is that in biology the more flexible notion of fibration symmetry is more natural and more useful. A fibration can be viewed as a system of ‘local’ symmetries, relating nodes in any given fiber. The closure property no longer holds, so these new kinds of symmetries do not form a group. Instead, the fibration symmetries form a more general structure called a ‘groupoid’. We develop this viewpoint in more detail as the book progresses.

We can illustrate the difference between global and local symmetries using a graph resembling Fig. 2.1, with one edge removed to simplify the construction. This graph is shown in Fig. 2.6b. We compare and contrast it with Fig. 2.6a.

Figure 2.6a has a global permutation symmetry

$$\rho = \begin{pmatrix} 1 & 2 & 3 \\ \downarrow & \downarrow & \downarrow \\ 1 & 3 & 2 \end{pmatrix}$$

which leaves node 1 fixed and interchanges 2 and 3. This permutation is a global symmetry of the entire network. Geometrically, it reflects the graph left-right. If we color symmetrically

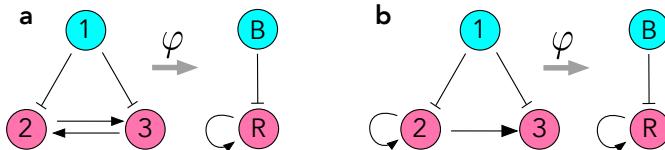


Fig. 2.6

Global and local symmetries. (a) This graph has global left-right symmetry.

Assigning the same color to symmetrically related nodes defines a fibration φ to a graph with two nodes. (b) This graph has no (nontrivial) global symmetry. However, assigning colors as shown also defines a fibration φ to a graph with two nodes.

related nodes with the same color, node 1 is colored blue while nodes 2 and 3 are colored red. There is now a fibration φ to a 2-node base, as shown.

Figure 2.6b has no global permutation symmetry (aside from a trivial symmetry: the identity permutation). If we apply the permutation ρ , we obtain a different graph, so ρ is not a global symmetry. Despite that, there is a fibration with the same coloring as in (a). The reason is that there is a local symmetry: nodes 2 and 3 receive exactly the same sets of input arrows, of the same types and with the same colors for source nodes. That is, the coloring is balanced. This is a local condition since it affects only the in-neighborhood of nodes 2 and 3, but not the rest of the graph, like node 1 and the outputs of 2 and 3.

The same phenomenon occurs in Fig. 2.1, which is why that graph has a nontrivial fibration symmetry, despite the lack of any global symmetry.

These examples illustrate two key points. First, we see that:

Balanced colorings (equivalently, symmetry fibrations) need not be determined by global (group) symmetries.

In full generality, it can be proved that:

Every balanced coloring is given by the fibers of a fibration symmetry, and conversely.

Since we have already seen that robust synchrony patterns correspond precisely to balanced colorings, it is clear that fibration symmetry is *the* central concept in network synchronization (here, robust, as defined in Section 5.9, is the key word):

Every *robust* cluster synchrony pattern is given by the fibers of a fibration symmetry, and conversely.

2.7 Physics and biology

The previous sections have introduced, in an informal manner, several basic concepts related to synchronization in networks. We end this chapter by summarizing some of the

most important implications for biological networks. The reasoning behind these statements will be explained in more detail as the book progresses.

In the physical sciences, the deepest features of the laws of physics are their symmetries. These symmetries are global, defined by a symmetry group; they occur because the laws of physics are the same everywhere and are compatible with group symmetries, which are rigid. For example, electrons are the same in all molecules; they are interchangeable and identical.

Biological sciences do not work like that. It is not even clear whether analogous ‘laws of biology’ exist, but if they do, they must explain how variability and flexibility lead to robust (in a non-technical sense) biological processes, in spite of various external and internal changes. For example, DNA is in every single cell and works in the same way, yet each species, even each organism, has a different genome. All electrons are the same, but every elephant is different (Wuppuluri and Stewart, 2022). They belong of the same species (more precisely, several closely related ones), but they are not identical.

This suggests that the group concept of symmetry used so successfully in physics is not entirely appropriate for biology. Group symmetries are too rigid. In particular, they are easily destroyed by any perturbation of network structure. It is reasonable to ask whether there might be some more general and more flexible notion of symmetry that is appropriate for biology. Unlike electrons, elephants are not identical, but they are easily distinguishable from mice or cats. This is some kind of symmetry statement, in a metaphorical sense, at least.

Can the metaphor be formalized? If so, is it useful?

In this book we argue that fibration symmetries provide just such a notion, at least for biological networks. The appropriate algebraic analog of a group turns out to be a groupoid, which can be viewed as a coherent system of local groups. Group symmetry preserves the global structure of a physical system. Groupoid symmetry does not preserve all relevant properties of a system, but it does preserve the system’s synchronized dynamics. A key feature of the groupoid structure is the associated notion of a fibration, so we also use the term ‘fibration symmetry’. Fibration symmetries are flexible and robust. And biology needs to be flexible and robust. It is therefore natural to seek to relate fibration symmetries to biology.

In this book, we use symmetry in this more general sense. The ‘correct’ notion of symmetry for biology, we contend, is fibration symmetry. The distinction is important, but there are some useful analogies, which help to motivate and organize the theory. For instance, in a network with group symmetry, symmetrically related objects are in the same orbit of the group. In a network with fibration symmetry, symmetrically related objects are in the same fiber. A group symmetry is always a fibration symmetry (but not the opposite). Every group of automorphisms has a corresponding fibration symmetry, whose fibers are the group orbits, but the converse need not hold. Thus fibration symmetries are strict generalizations of automorphisms. Their local nature makes them more flexible and more robust. Small changes to a network can destroy group symmetry completely. But many of the fibration symmetries remain.

Part I of the book is mainly directed towards setting up and explaining the corresponding mathematical machinery, although we use biological examples when we can. All very well,

but ‘the proof of the pudding is in the eating’. The proof that fibration symmetries are useful in biological networks requires us to put all this machinery into action, to solve interesting biological problems. Part II of this book is an attempt to do exactly that.

Graphs, Networks, and Synchronization

We now begin to supply a little more detail and to define concepts in precise mathematical terms. We continue to illustrate the formal concepts with examples, and avoid mathematical complexities where possible. This chapter defines the basic concepts that will be used in this book, such as the difference between a graph and a network (and the admissible dynamical systems of equations defining its dynamics), the notion of partitions of a graph, and the difference between complete and cluster synchronization. We also elaborate on an important problem treated in this book: the structure-function relation in networks.

3.1 Basic definitions

To set the scene, we consider a general biological system composed of N interacting units, such as neurons or genes. The dynamical behavior of each biological unit is described by a continuous variable $x_i(t)$, which measures the biological activity of unit i at time t , such as the firing of a neuron (Hodgkin and Huxley, 1952) or the gene expression level of mRNA and proteins (Klipp et al., 2016; Alon, 2019). We use a network representation of biological systems (Buchanan et al., 2010), where biological units are represented as nodes, and their interactions as directed links between nodes. The nodes and links form a *biological graph*, and when we attach a dynamical system to every node they form a *biological network (system)*.

Definition 3.1 **Directed graph** (Harary, 1969; Berge, 1973). A *directed graph* $G = (V, E)$ is defined by a set $V = \{1, \dots, N\}$ of $N = |V|$ nodes (vertices, cells, compartments, units...) and a set $E \subseteq V \times V$ of $M = |E|$ edges (arcs, links, arrows). The set E determines the connections among the nodes. If $E = \{e_1, e_2, \dots, e_M\}$, we can write each edge in the form $e_\ell = (i, j)$, with $\ell \in [1, M]$. This pair of nodes represents a directed connection from node $i \in V$ (the *source* of the edge) to node $j \in V$ (the *target* of the edge). The source is denoted by $s(e_\ell)$ and the target is denoted by $t(e_\ell)$.

Remark In various parts of the literature, the source (also called *tail*) is denoted by $\mathbf{t}(e_\ell)$ or $\mathcal{T}(e_\ell)$. The target (also called *head*) is denoted by $\mathbf{h}(e_\ell)$ or $\mathcal{H}(e_\ell)$. We do not use these alternative notations.

The terms ‘node’ and ‘link’ are usually used in the complex network literature (Buchanan et al., 2010), while ‘vertex’ and ‘edge/arc’ are employed in the graph theory literature (Harary, 1969; Bollobás, 2012) especially when referring to undirected networks (i.e., networks where arc direction is irrelevant). The terms ‘node’ and ‘arrow’ are used in

(Golubitsky and Stewart, 2023) and related articles. In this book, we mainly use ‘node’ and ‘edge/link’, but ‘vertex’ and ‘arc’ are also used, depending on context.

Definition 3.2 Adjacency matrix of a directed graph. Let $G = (V, E)$ be a directed graph. The *adjacency matrix* A of the graph G is a $N \times N$ matrix ($N = |V|$) with elements A_{ij} representing the connectivity of the graph:

$$A = \begin{cases} A_{ij} = 1 & \text{if there exists an edge } e_\ell = (i, j) \\ A_{ij} = 0 & \text{otherwise.} \end{cases} \quad (3.1)$$

The edge (i, j) represents an *output* from node i and an *input* to node j .

We write $G(A)$ for the graph determined by the adjacency matrix A .

Definition 3.3 Undirected graph. A graph $G = (V, E)$ is *undirected* if and only if $(i, j) \in E$ implies $(j, i) \in E$. The pair of arcs $(i, j), (j, i)$ is interpreted as a single undirected connection between i and j , which is called an *edge*. In this context, nodes are often referred to as *vertices*.

In other words, an undirected graph can be thought of as a special type of directed graph, in which those pairs of nodes that are connected are linked by two directed edges, one in each direction.

We shall use the directed and undirected terminologies interchangeably, and the context will usually be enough to distinguish whether we are talking about a directed or an undirected graph. In particular we often use ‘edge’ rather than ‘arc’ for directed graphs.

Undirected graphs are also called ‘symmetric’ graphs, but we do not use this name to avoid confusion.

A graph may be weighted or unweighted. In a *weighted* graph each edge e_ℓ is associated with the *weight* $\omega_\ell \in \mathbb{R}$. For an unweighted graph all interactions are weighted equally and $\omega_\ell = 1$.

The adjacency matrix of a weighted (directed or undirected) graph is defined in the same way as that of an unweighted graph, but now we set $A_{ij} = \omega_{ij}$, the weight of the corresponding edge from i to j .

Sometimes, edges can have a type (like in the example of Fig. 2.1, where we have repressor and activator edges). Similarly to weights, types provide extra information about the nature of the corresponding edges. Also nodes can have a type: this is especially important for networks that include nodes with different classes (for instance, proteins, genes, metabolites etc.).

Definition 3.4 Node/edge typing. A *typed graph* is a graph $G = (V, E)$ endowed with a set T of types, and two functions $\nu : V \rightarrow T$ and $\eta : E \rightarrow T$ that assigns a type to every node and edge respectively.

Although we are using one single set of types, normally the types of nodes and edges are different. In the following, we can assume that all nodes have the same type, and the same for edges, unless otherwise specified. Some more systematical examples of graphs with different types of nodes and edges is postponed to Section 9.5.

Warning: Some authors, such as (Golubitsky and Stewart, 2023), define the adjacency matrix so that A_{ij} represents edges from node j to node i ; that is, they use the transpose of the adjacency matrix defined above. This modification is slightly more natural when constructing admissible equations, but to avoid possible confusion we do not use it in this book.

It is also important to define subgraph types since they are useful for defining building blocks.

Definition 3.5 Subgraph (Harary, 1969). A *subgraph* $G' = (N', E')$ of a graph $G = (N, E)$ is a graph such that $N' \subseteq N$ and $E' \subseteq E$, and if $e \in E'$ then $s(e), t(e) \in N'$.

That is, G' is a subgraph of G if G' consists of some nodes and some edges of the nodes and edges of G , and the sources and targets of edges are among the nodes concerned.

Definition 3.6 Induced subgraph (Harary, 1969). An *induced subgraph* $G' = (N', E')$ of $G = (N, E)$, induced by the vertex set $N' \subseteq N$, is the graph $G' = (N', E')$ such that $E' = \{e = (n_1, n_2) \in E \mid n_1, n_2 \in N'\}$. The subgraph of G induced by $N' \subseteq N$ is denoted by $G[N']$ and contains *all* of the edges in E that connect nodes in N' .

The difference between an induced subgraph and a subgraph is that an induced subgraph includes all the edges connecting the nodes in it, while a subgraph can omit some edges. This definition will be useful when defining the building blocks of the network in Part II, since such a circuit can be studied either in isolation from the original graph or embedded on it.

We end this section with two further concepts that we use throughout the book:

Definition 3.7 Directed path. A *(directed) path* in a directed graph is any sequence of directed edges such that the target/head node of each edge is the source/tail node of the next. Repetitions of nodes or edges are permitted.

In graph theory the term ‘walk’ is often used for this concept, a ‘trail’ is a walk with no repeated edges, and a ‘path’ is a walk with no repeated vertices. In this book, we use ‘path’ rather than ‘walk’. SCCs will be used in Part II in the analysis of biological networks.

Definition 3.8 Strongly connected component (SCC). A directed graph is *strongly connected* if there is a (directed) path between all pairs of vertices. A *strongly connected component* (SCC) of a directed graph is a maximal induced subgraph that is strongly connected.

For brevity, we often omit the adjective ‘strongly’ when this does not cause confusion, and talk of ‘connected components’ of a directed graph. In the graph theory literature this term usually refers to an analogous concept based on undirected paths, but we do not need that concept in this book since it is not related to network dynamics in any useful manner.

3.2 Graphs, multigraphs, and hypergraphs

Sometimes we need to allow graphs to have multiple edges between a given pair of nodes, in part because in some cases a fibration can collapse a graph without multiple links to a base that has multiple links (Golubitsky and Stewart, 2023, Sections 8.10, 10.6). This is already apparent in Fig. 1.3. The graph G has no loops or multiple edges, but the base B has both. These features are important to make the admissible maps for the base correspond precisely to synchronized restrictions of admissible maps for the graph.

To allow for multiple edges we need a slightly more general notion of graph:

Definition 3.9 Multigraph (Berge, 1984). A *multigraph* $G = (V, E, s, t)$ consists of a set of nodes V and a set of multiedges E allowing for multiple edges (parallel edges) between two nodes (while a graph allows only one edge connecting any two given nodes). The two functions $s, t : E \rightarrow V$ describe the source and the target node of each edge: the directed edge $e \in E$ has source $s(e)$ and target $t(e)$.

A graph can always be seen as a multigraph in which each edge has multiplicity 1, so the notion of a multigraph is in fact an extension of Definition 3.1. The notions of edge weight and/or type carry over to multigraphs.

In the following, it is convenient to use the term ‘graph’ for either a graph in the strict sense or a multigraph, and we do this when it is clear which concept is intended; in this case, we say *simple graph* to mean a graph according to Definition 3.1.

In a simple graph, two nodes uniquely define one edge, so we can use the simplified notation $e_\ell = (i, j) \in E$, which is equivalent to letting $s(e) = i$ and $t(e) = j$. This notation is unambiguous because in a simple graph the source and target nodes uniquely define the edge, which is not the case for a multigraph.

It is sometimes asserted that since the network formalism of (Golubitsky and Stewart, 2023) is based on graphs and multigraphs, it describes only two-body interactions. On the contrary, the dynamic of a node depends on *all* of its input nodes, and can be any function of their states; it is not limited to a linear combination of pairwise interactions. This misconception may have arisen through confusion with standard ‘linear interaction’ models in which the interaction term is additive in the input variables and the interaction strengths are prescribed by an adjacency matrix.

The main issue here is not whether many-body interactions are permitted, but what extra constraints can be imposed on them. A systematic way to do this is to use hypergraphs (Berge, 1984; Von der Gracht et al., 2023), which we discuss in Chapter 9.

Most of the examples in the book are treated with simple graphs and multigraphs unless otherwise noted. Multigraphs are important since the application of a fibration to a graph usually leads to a graph with multiple parallel edges. Hypergraphs are treated in Section 9.2. Examples of the different types of graphs can be seen in Fig. 3.1.

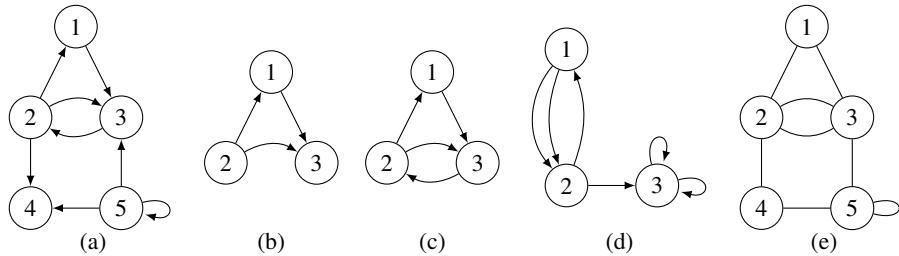


Fig. 3.1

Simple examples of graphs. (a) A graph G with 5 nodes and 8 edges (one of them is a self-loop on node 5). (b) A subgraph of G . (c) An induced subgraph of G (the subgraph induced by $\{1, 2, 3\}$). (d) A multigraph. Node 3 has two loops, and there are two parallel arcs from 1 to 2. (e) An undirected multigraph (every edge should be interpreted as a pair of arcs in opposite directions; the loop itself is just a loop).

3.3 Setting up model equations

In many areas of science, networks are used to set up model equations. Typically, nodes represent state variables and edges represent couplings. The network structure is a modeling assumption, which generally represents a simplification of a more complex reality. As we said at the beginning of this section, each node i in the graph comes with its state variable $x_i(t)$ (the state of node i at time t), which is an element of some node space M_i .

The model equations are often selected from a small catalog of equations that are standard in each specialized area (Buchanan et al., 2010)—for example, in neuroscience, typical model neurons are governed by FitzHugh–Nagumo (FitzHugh, 1961; Nagumo et al., 1962), Morris–Lecar (Morris and Lecar, 1981), or Hodgkin–Huxley (Hodgkin and Huxley, 1952) equations. In systems biology, Hill functions are used to model binding interactions between protein and DNA in gene regulatory networks (Alon, 2019), while Michaelis–Menten kinetics describe biochemical reactions in metabolic networks (Klipp et al., 2016). Couplings are also standardized, for example as additive voltage coupling or multiplicative gates in gene regulation (Alon, 2019).

The manner in which the network embodies the model equations often differs between areas of application. An example is the distinction between activator and repressor edges, with (fairly) standard symbols for each. In molecular reaction networks, edges may point to other edges, not nodes. When conservation laws such as mass balance apply, an edge sometimes indicates an output from its source as well as an input to its target. These conventions must be borne in mind when comparing network diagrams in different areas. The network diagram may then take on a different appearance from that assumed in this book, which represents the structure of the *influence network*: which node variables affect which other node variables.

3.4 Admissible ODEs

Although the models are standard in a given area, they differ from one area to another. Moreover, the model may be an ordinary differential equation (ODE), a delay differential equation, a stochastic differential equation, or have other (for instance, probabilistic) features. Further, the dynamic can be continuous time (differential equation) or discrete time (iterated map). A general theory has to encompass this variety of models, but it also has to avoid being *too* general. We therefore, restrict attention mostly to ODE models, although iterative maps will also be treated in few occasions. Because of the variety of models encountered in different areas, a key feature of the general formalism is that a graph determines an entire *class* of differential equations:

Definition 3.10 Admissible differential equations (Stewart et al., 2003; Golubitsky and Stewart, 2006, 2023). An ODE is *admissible* for a graph G if it respects the graph structure of G . That is, the component of the ODE for node c has the form

$$\dot{x}_c = f_c(x_c, x_{i_1}, \dots, x_{i_k}) \quad (3.2)$$

where the i_j are the source nodes of the input edges to c . The function f_c is called the *input function* at node c , and it specifies how the inputs $(x_{i_1}, \dots, x_{i_k})$ are combined to determine the dynamics at node c . The first variable in f_c is the node c , specifying the internal dynamics of the node. Moreover, if nodes c and d have isomorphic input sets then $f_c = f_d$ (the ordering of the input edges needs to be consistent with the isomorphism). This means that f_c and f_d represent the same input functions (same type and same coefficients defining the interaction edge).

The admissible ODEs are not specific equations; they define the class of *all* ODEs that respect the graph structure. The specific equations that are standard in any given area belong to the class of admissible equations—provided they are ODEs—so general results can be specialized to those models. However, special models can have additional features, not present in every admissible equation. Examples of such features are the mass-balance condition in chemical kinetics, diffusive or generalized diffusive coupling, and the widespread use of linear coupling.

The definition of admissible equations generalizes readily to other kinds of dynamic: discrete time dynamics (iterated functions), finite-state automata, Boolean networks, and so on, although a general theory is currently less well developed in these contexts.

3.4.1 From graph to admissible ODEs

The concept of admissible equations assumes that the equations of the dynamical system respect the structure of the graph. In particular, the symmetries (group or fibration) of the graph are respected by the dynamical system too.

The admissible equations are a set of all models compatible with the graph. However, most of the time, we are interested in a particular ODE that represents the biological system

of study. In this case, there is a one-to-one relation between the graph or the hypergraph and the ODE: we can write the form of the dynamical equations using only the information contained in the graph. Each edge in the graph represents one type of coupling or interaction variable in the dynamical system, and the information contained in each edge is enough to specify the form of the dynamical equation (modulo some general considerations) It is also necessary to choose the node spaces M_i : the structural form of admissible ODEs is the same for all such choices, but the meaning of the node coordinate x_i differs. Examples of these are discussed in Section 9.3.

Definition 3.11 Network dynamical system (Stewart et al., 2003; Golubitsky and Stewart, 2006). We define a *network dynamical system* corresponding to the graph $G = (V, E)$ by attaching to every node $i \in G$ a *state* determined by a variable $x_i(t) \in M_i$ taking values in a finite-dimensional vector space M_i . In most cases treated here we use $M_i = \mathbb{R}$. Moreover, the state $x_i(t)$ evolves following the set of admissible equations for the graph under the influence of the nodes specified by the graph.

Examples of biological networks defined on an underlying graph, multigraph or hypergraph are:

- The transcriptome, which encompasses the activity of all RNA transcripts interacting on an underlying transcriptional regulatory graph.
- The proteome, which encompasses the activity of all proteins expressed in the cell interacting on an underlying protein interaction network.
- The metabolome, made of small molecule chemicals and metabolites interacting in a metabolic graph.
- The neural dynamical network of brain activity produced by neurons interacting in the underlying graph connectome.

These networks are coupled via other biological processes, such as signaling pathways and the environment. The functionality of the cell then emerges at the system level through the integration of these basic ‘omics’ networks.

Conversely, the form of admissible ODEs specifies the connections and distinguishes their types, with two caveats. First, in the general network theory we can say when two edges have the same types, or different types, but not what those types are. All we know is when they are the same and when they differ. Two nodes whose input sets are in a one-to-one correspondence that preserves types are ‘input isomorphic’, and the correspondence is an ‘input isomorphism’. Second, if two nodes are *not* input isomorphic, there is no formal relation between the input edges of one node and those of the other. The next example illustrates these statements. Definition 7.1 provides a formal definition.

Example 1 Consider the network of Fig. 2.1. Admissible ODEs have the form

$$\begin{aligned}\dot{x}_1 &= f(x_1, x_2) \\ \dot{x}_2 &= g(x_2, x_1, x_2) \\ \dot{x}_3 &= g(x_3, x_1, x_2)\end{aligned}\tag{3.3}$$

for arbitrary input functions f, g with suitable domain and codomain.

We now explain why that form follows from (3.2), i.e., it is admissible for the graph. As discussed in Section 2.3, the input sets of the nodes are:

$$\begin{aligned} \text{Node 1 : } & \{e_2\}. \\ \text{Node 2 : } & \{e_1, e_4\}. \\ \text{Node 3 : } & \{e_3, e_5\}. \end{aligned} \tag{3.4}$$

We see that node 1 is not input isomorphic to the other two nodes, because its input set consists of a single edge, whereas nodes 2 and 3 have two input edges. Nodes 2 and 3 are input isomorphic to each other: the correspondence

$$e_1 \leftrightarrow e_3 \quad e_4 \leftrightarrow e_5$$

is one-to-one and corresponding edges have the same type. In this case, e_1 and e_3 are inhibitors, while e_4 and e_5 are activators.

Therefore we use a function $f_1 = f$ for the \dot{x}_1 equation, and a *different* function $f_2 = f_3 = g$ for the \dot{x}_2 and \dot{x}_3 equations. The fact that $f_2 = f_3 = g$ is important. We can see from the graph that they both represent an input function receiving one inhibitor edge (from node 1) and one activator edge (from node 2). Thus, f_2 and f_3 are the same input functions. However, a further (somehow hidden) assumption is that these input functions have the same coefficients defining the interaction terms. We will come back to this issue below and in Chapter 21.

The behavior of node 1 depends on its own state x_1 (this is indicated by the first variable in $f(x_1, x_2)$), and on that of the only node that is the source of its input edge e_2 , which is node 2 (indicated by the second variable in $f(x_1, x_2)$). In the \dot{x}_1 equation, we apply f to these node variables to obtain:

$$\dot{x}_1 = f(x_1, x_2).$$

The behavior of node 2 depends on its own state x_2 , and on that of the two nodes that are the sources of its input edges e_1, e_4 , which are nodes 1 and 2 respectively. In the \dot{x}_2 equation, we apply g to these node variables:

$$\dot{x}_2 = g(x_2, x_1, x_2).$$

The behavior of node 3 depends on its own state x_3 , and on that of the two nodes that are the sources of its input edges e_3, e_5 , which are nodes 1 and 2 respectively. In the \dot{x}_3 equation, we apply g to these node variables:

$$\dot{x}_3 = g(x_3, x_1, x_2).$$

Here, it is important to list the variables for nodes 2 and 3 in the order that gives the input isomorphism, which, in this case is: node, repressor, activator.

The result is the form of admissible ODEs stated in (3.3).

3.4.2 From admissible ODEs to graph

We can also ‘reverse-engineer’ the network from the form of admissible ODEs, subject to some technical remarks below. Since there are three equations, there are three corresponding

nodes. The variables inside the function on the right-hand side specify the source nodes of input edges. By convention, the first variable represents the node itself. For node 1 there is only one input edge, with source 2, so there is an edge from 2 to 1. Similarly, the variables in the equations for nodes 2 and 3 list the sources of their inputs in corresponding order. One subtlety of the general theory is that when a node has several input edges of the same type, the function is symmetric under all permutations of those edges. (These permutations are input *automorphisms*; that is, isomorphisms of the input set with itself; the form of the ODE must be invariant under all input isomorphisms, and that includes the input automorphisms. Another term is *vertex symmetry*.) Since we have not specified any such symmetries for the function g , the two input edges are of different types.

A further subtlety is that although admissible ODEs distinguish different types of edge, these types do not themselves tell us which are repressors and which are activators. This level of detail comes into play only when we specialize the ODE to more specific models. Similarly, because node 1 is not input isomorphic to nodes 2 and 3, the admissible ODE does not tell us that node e_2 is ‘the same as’ nodes e_1 and e_3 , so it does not specify that this edge is a repressor. Again, that information must be provided by specializing the ODE model.

It is common to associate individual *terms* in a model ODE with the edges (and with the nodes). Often, inputs are combined by adding them (or, in some cases, multiplying them). In such cases, the model also contains information on the type of edge—say using a positive coefficient for an activator and a negative one for a repressor. However, in other areas, this kind of additive structure for the inputs is not appropriate, which is why the general theory does not assume it. In general, inputs can be combined in any form, representing a multi-body interaction that can be described in more detail by a hypergraph if desired (Section 9.2). The general form (3.2) is valid for graphs and hypergraphs.

If there exist different types of interactions between nodes, this should be reflected in the graph. For instance, genetic networks are composed of activators and repressors, and neural networks contain excitatory and inhibitory interactions, so two classes of edges are required to describe these networks. When the network is unweighted and untyped, all edges are the same. However, when the edges are weighted/typed, representing, for instance, different strength/type of interactions, then each edge should, in principle, come with its specific weight/type. We treat these cases in detail in Chapter 21.

The question arises: why do we not use a hypergraph to describe the network in Fig. 2.1? After all, nodes 2 and 3 receive two inputs each, and we should specify how these inputs are combined through input functions specified by a hypergraph. However, the key assumption is that both nodes combine their inputs in the same way, i.e., they are input isomorphic. This is why we use the same function g to specify the input functions at both nodes. Additionally, it is biologically plausible—though not true in general—that edges e_1 and e_3 represent the same interaction terms in the ODE (not only the same type, like activator/inhibitor but also with the same coefficients defining the input function, eg, the same Hill function with the same coefficient and combined in the same way) since they originate from the same source node (the same applies to edges e_4 and e_5). Therefore, the complexity of the hypergraph representation can be effectively simplified to a standard graph structure.

However, if the input functions at these nodes differ, then the same function g cannot be

used for 2 and 3, and we may need to employ a hypergraph to accurately describe how the inputs are combined at each of these nodes. This discussion may sound a bit cryptic at this point, but it will make sense when we apply the concepts to specific biological networks in Part II; we dedicate Section 9.3 to an analysis of this issue.

3.5 Partition of a graph

To understand important concepts in this book such as cluster synchronization, orbits, fibers and balanced colorings/equitable partitions, it is useful to introduce the concepts of a partition and the corresponding graph coloring. Formally:

Definition 3.12 Partition of a graph. Let $G = (V, E)$ be a graph with $N = |V|$ nodes. Let c_1, \dots, c_K be subsets of nodes. Then $\mathcal{P} = \{c_1, \dots, c_K\}$ is a *partition* of the graph (and the sets c_k are the *parts* or *clusters* of the partition \mathcal{P}) if $\bigcup_{k=1}^K c_k = V$, $c_k \neq \emptyset$, $c_k \cap c_l = \emptyset$ for $k \neq l$. Clearly, $N = \sum_{k=1}^K |c_k|$ where $|c_k|$ is the number of nodes in cluster c_k .

In other words, \mathcal{P} partitions the graph into non-overlapping (or disjoint) non-empty sets of nodes, c_k , which are the clusters (or parts) of the partition. If $|c_k| = 1$, then the cluster is *trivial*; *nontrivial* clusters contain more than one node. A useful way to visualize a partition is to assign a color to each cluster, creating a colored graph.

Definition 3.13 Colored graph. A (*node-*)colored graph with K colors $\{\phi_1, \phi_2, \dots, \phi_K\}$ is a pair (G, ϕ) comprising a graph $G = (V, E)$ and a function $\phi : V \rightarrow \{\phi_1, \phi_2, \dots, \phi_K\}$ that maps nodes of G to their colors.

Intuitively, this definition provides a convenient way to visualize the partition. Clearly, the two notions are equivalent: every partition into K clusters defines a coloring (using K colors, and coloring two nodes with the same color if and only if they belong to the same cluster); and every coloring can be interpreted as a partition (which gathers all nodes with the same color in the same cluster). We are here assuming, without loss of generality, that ϕ is surjective (i.e., that all colors are used).

We have not imposed any restrictions or constraints on the assignment of colors. Imposing constraints creates different types of *coloring problems*, like the balanced colorings/equitable partition to be treated in Chapter 5, or the famous (and unrelated to balanced coloring) *vertex coloring problem* which consists of coloring the vertices of an undirected graph in such a way that no two adjacent vertices have the same color. In what follows, we consider two main types of partitions/colorings of the graph: orbits, discussed in section 4.2, and balanced colorings/equitable partitions/fibers, which are equivalent to each other and are discussed in Chapter 5. These partitions are related to each other, and each leads to cluster synchronization in its own way.

3.6 Complete synchronization and cluster synchronization

Drawing a conclusion about network dynamics, and in particular synchronization, based on the symmetries of the graph is a wide field of research. The problem of generic synchronization can be studied for the general set of equations (3.2). It is important to distinguish between complete synchronization and cluster synchronization.

Definition 3.14 Complete synchronization. *Complete synchronization* for the system of equations (3.2), admissible for a graph $G = (V, E)$ with $N = |V|$ nodes, arises when

$$x_1(t) = x_2(t) = \dots = x_N(t). \quad (3.5)$$

Such a state is *completely synchronous* or *completely synchronized*.

A network is *completely synchronizable* if some open set of initial conditions converges to a completely synchronous state. That is, there is a dynamically stable, completely synchronous state.

Complete synchronization refers to the *existence* of completely synchronous solutions, while complete synchronizability refers to the ability of the network to achieve a *stable* completely synchronous state (Pecora and Carroll, 1998).

Synchronization and synchronizability have been extensively studied in the engineering, nonlinear dynamics and physics communities (Belykh et al., 2008; Do et al., 2012; Dahms et al., 2012; Fu et al., 2014; Kanter et al., 2011; Pecora and Carroll, 1998; Rosin et al., 2013; Sorrentino and Ott, 2007; Strogatz, 2018). Some of these papers consider the more general phenomenon of cluster synchronization, which is a collective behavior of dynamical systems in which only certain subsets (modules, clusters) of nodes undergo the same evolution. Following (Stewart et al., 2003; DeVille and Lerman, 2015b; Nijholt et al., 2016; Sorrentino et al., 2016; Siddique et al., 2018) we define cluster synchrony as follows:

Definition 3.15 Cluster synchronization. *Cluster synchronization* for the system of equations (3.2), admissible for a graph $G = (V, E)$, arises when there exists a graph partition $\mathcal{P} = \{c_1, \dots, c_K\}$ such that

$$x_i(t) = x_j(t) \text{ for } i, j \in c_k, \quad k = 1, \dots, K. \quad (3.6)$$

In the language of (Stewart et al., 2003), cluster synchronization defines a *synchrony subspace* or *polydiagonal subspace*

$$\Delta_\phi = \{x : x_c = x_d \text{ whenever } c \text{ and } d \text{ are synchronized}\}.$$

Another term is *cluster synchronization manifold*. The key feature is that for balanced colorings, and only those colorings, this subspace is invariant under any admissible ODE. Intuitively, this means that nodes that initially lie in the same cluster remain in the same

cluster as time passes. (However, they may not do so stably. Perturbations may destroy the synchrony pattern.) The dynamics of the full network, when restricted to the synchrony subspace, represent the dynamics of the clusters.

3.6.1 Laplacian systems

In the graph theory literature, the Laplacian matrix of a graph is at least as important as that of the adjacency matrix. Indeed, a huge amount of work on synchronization in networks has been carried out for Laplacian systems. We, therefore, give a brief and highly simplified description of this approach, and then explain why we do not adopt it in this book.

The *Laplacian (matrix)* L of a graph (simple, undirected, and unweighted) is defined by:

$$L_{ij} = \begin{cases} \deg(i) & \text{if } i = j \\ -1 & \text{if } i \neq j \text{ and } i \text{ is adjacent to } j \end{cases}$$

where i, j runs through the nodes. Here, \deg is the in-degree, which equals the number of adjacent nodes. The Laplacian can also be written as $L = D - A$, with D the degree matrix and A the adjacency matrix.

Suppose that the dynamic of a network is defined by diffusive couplings, meaning that all interactions are additive and of the form $H(x_i) - H(x_j)$, for some possibly nonlinear function H , with the same H for every node. Then the interaction between synchronous nodes, where $x_i = x_j$, is $H(x_i) - H(x_i)$, which is zero. That is, synchronous pairs of nodes *decouple*. Typically, the linear part of the model ODE is then determined by the Laplacian.

Assume that all nodes have the same state space, and that the internal dynamics of all nodes are identical: $\dot{x}_i = F(x_i)$ for a fixed F . Then, the model ODE takes the form

$$\dot{x}_i = F(x_i) - S \sum_{j \neq i} A_{ij}(H(x_i) - H(x_j)), \quad i = 1, \dots, N, \quad (3.7)$$

where A is the adjacency matrix and S a coupling constant.

Assuming $H(x)$ is linear, we obtain ‘Laplacian’ dynamics

$$\dot{x}_i = F(x_i) - S \sum_{j \neq i} L_{ij} H(x_j), \quad i = 1, \dots, N. \quad (3.8)$$

(See Section 8.10 for more detail.) Interactions of the form (3.7) and (3.8) imply that a completely synchronized state *always* exists, no matter what the topology of the graph may be (Pecora and Carroll, 1998). The reason is that synchronous nodes decouple dynamically for this kind of interaction, unlike more general couplings. To see why, start with a completely synchronous initial condition $x_i(0) = x_j(0)$. Then each node satisfies the same ODE $\dot{x}_i = F(x_i)$ with the same initial condition on $x_i(0)$, so $x_i(t) = x_j(t)$ for all times t . In other words, the *existence* of a completely synchronous state is built into the model from the start. From the general viewpoint, the coloring with all nodes having the same color is not balanced. This can be dealt with by introducing a notion of balance tailored to such models: ignore input arrows whose source and target have the same color.

Whether this synchronous state is *stable* is another issue, and much work has been done on that question; see Section 8.10, which explains how the Laplacian enters into the stability

analysis. Similar remarks apply to another widely used type of model, the Kuramoto model (Kuramoto, 1984).

In this book, we assume more general interactions than those described by diffusive couplings and Laplacians. The existence of a *completely* synchronous state then depends on the topology of the network and often does not exist. Cluster synchrony may occur even when complete synchrony does not, which is why we focus on clusters.

Moreover, as discussed in section 2.1, few systems in biology have interaction terms like $H(x_i) - H(x_j)$ (a notable exception being Turing-type reaction-diffusion models of pattern formation and development), and complete synchronization is not common (and deleterious) in biological networks. In particular, none of the biological systems studied in this book is diffusive. Thus, these systems are not described by the Laplacian L but by the adjacency matrix A . This remark includes genetic, protein, metabolic, signaling, ecological, and brain networks, as well as artificial neural networks.

For Laplacian models, the link between synchrony and fibrations changes dramatically. For example, a different notion of a balanced coloring is required to characterize robust synchrony patterns, which presumably requires a different notion of a fibration. The main reason for this difference is that in the Laplacian case, the dynamics of a given node is unaffected by any nodes that are synchronous with it. So inputs to such a node from any synchronous nodes should be ignored when seeking robust synchrony patterns. Moreover, which nodes should be ignored depends on the pattern.

For this reason, despite their importance in other areas, we do not address Laplacian-based models again, except in Section 8.10.

3.6.2 Cluster synchronization and cellular function

An important question addressed in this book is: which cluster synchronization and partitions are supported by the different symmetries of the graph? In general, there may be many such partitions, and enumerating them all is computationally challenging (Kamei and Cock, 2013a). We discuss cluster synchronization by automorphisms in Chapter 4, and by symmetry fibrations in Chapter 6. Algorithms to find fibrations, hence clusters, are described in Chapter 13.

Cluster synchronization is important because different synchronized clusters of units in a biological network perform different biological functions. In this scenario, biological structures can be pictured as an orchestra in which each instrument is a node. (However, unlike a real orchestra, there is no conductor—unless that role is played by the flow of time.) When the instruments play coherently, with synchronized, rhythmical or structured temporal patterns, the system becomes functional. Here, we concentrate on the simplest temporal organization, one in which some units (instruments) act synchronously in time in clusters (modules), a ubiquitous pattern observed in all biological systems.

Biological systems are constrained by the requirements of modularity and integration (Hartwell et al., 1999a). Cellular functions are performed by modules: segregated molecular units synchronize to perform a specific function in the cell. Modules ought to be sufficiently independent to guarantee functional specialization and, at the same time sufficiently connected to bind multiple units for efficient global integration that guaran-

tees higher-level emergent behavior of a unified cellular machine (Hartwell et al., 1999a). How to realize the conflicting requirements of modularity in the presence of integration is a conundrum that has been largely studied in neuroscience (Tononi et al., 1994), but it applies equally to any information-processing living system, from genes to proteins to organisms (Hartwell et al., 1999a).

Modules can perform independent synchronized functions, yet they can still be integrated in the network, providing segregation of function in the presence of global integration. Biological cluster synchronization allows functional modules of biomolecules, proteins, metabolites, neurons and other features to work together in coherent dynamics. This includes clusters of co-expression patterns of genes and the cluster of synchronized neural ensembles in the brain (Strogatz, 2018; Klipp et al., 2016). Thus, cluster synchronization is required by a functional biological network, while complete synchronization is not.

For weighted (and unweighted) networks, with *linear* coupling, the way to obtain complete synchronization from a system of equations is to ensure that the rows of the underlying adjacency matrix A sum to a constant, which is analogous to using diffusive couplings in the linear regime, see (3.8). This leads to the use of the Laplacian matrix to describe the dynamics (Pecora and Carroll, 1998) explained in section 3.6.1. As discussed there, this assumption is not realistic for biological systems since interactions are not always diffusive, so they are not captured correctly by Laplacian matrices.

For a general network, complete synchronization is possible if each adjacency matrix has constant row-sums, because this condition means that all nodes have the same number of input edges of any given type. In this book we do not assume that there is a single adjacency matrix A whose rows all sum to zero. Removing this assumption (which effectively is the condition that $f(0) = 0$ in the model equation $\dot{x} = f(x)$) implies that complete synchronization may not be achievable. However, it is still possible for several synchronized clusters or modules to occur, as opposed to only one cluster corresponding to complete synchronization in the entire network. These clusters arise through fibrations, the main subject of this book.

3.7 From graph to network dynamics – from structure to function

In this book, we often distinguish between the terms ‘graph’ and ‘network’. ‘Graph’ is straightforward: a set of nodes and (usually directed) edges that satisfies Definition 3.1. In part of the literature, a ‘network’ is a graph whose nodes and edges can be split into distinct types (Golubitsky and Stewart, 2023, Chapter 9). Graph theorists call such graphs ‘colored graphs’, where the colors represent different types, but in this book we associate colors with clusters. We therefore avoid this terminology. A graph can be viewed as a network with only one type of edge, so the term ‘network’ includes ‘graph’.

When a graph is equipped with state variables and dynamical equations, it technically becomes a ‘network system’ of ODEs. However, we sometimes abbreviate this term to ‘network’ when it is clear that we are referring to an admissible equation. The point of view

is that a graph is just a combinatorial structure, whereas a network is a graph with extra features, intended to be used to define the structure of the associated differential equations.

To complicate matters, the terms ‘graph’ and ‘network’ are often used interchangeably in parts of the literature, especially in the field of complex networks, and we anticipate that we may also interchange them here from time to time. However, separating the graph from the network can be important, since this difference underlies a recurrent problem that will come up in this book many times: *how structure determines function* in biological systems. By structure we mean the structure of the graph, and by function we mean the function of the biological system given by the solution of an admissible dynamical system of equations of the corresponding network.

Clearly specific features of the dynamics, such as the quantitative value of a state variable or the frequency of an oscillation, depend on precise details of the model equations. We focus on more general features, which can occur for broad classes of models. Synchronization is the prime example of such a feature.

The types of questions we ask are:

- What can we say about the dynamics and function of a network based solely on the graph structure?
- Can we predict the function of a network or parts of it by knowing the structure of the graph alone, independently of the dynamical equations?
- What are the features of the graph that determine the function/dynamics of the network?
- Under what conditions can we decouple the dynamics from the structure?
- To be more concrete, can we predict anything about the dynamics, for instance, cluster synchronization, by knowing only the graph?

These are admittedly difficult questions at the core of systems science. They appear systematically in the study of biological networks like gene regulation and brain networks, and there are no easy answers for most of them. When the graph has no structure—think for instance of an Erdős–Rényi graph where nodes are connected at random—only limited conclusions can be drawn about its function. Away from randomness, many structures have been found in graphs, specially in biological graphs, such as modules, motifs, hubs, small worlds, fractality and others. However, most of these structures say little about the function of the network.

We will show that when the structure of the graph is determined by symmetries, in particular fibration symmetries, then important aspects of the relation between structure (graph symmetries and broken symmetries) and dynamics in the network (cluster synchronization and bifurcations) become clear. We develop this line of thinking throughout the book.

Automorphisms and Symmetry Groups

This book is about symmetry fibrations. But in order to understand these symmetries, we start with more familiar notions of automorphisms and symmetry groups. Traditionally, graph symmetries have been studied in terms of graph automorphisms, which are permutations of nodes that preserve the graph connectivity. For the didactic purpose of understanding fibrations in the context of known symmetries, we first discuss classical symmetries given by automorphism groups. We review these ideas as a warm-up, before introducing fibration symmetries and groupoids as the relevant symmetries in biology, as discussed in subsequent chapters. The reader familiar with group symmetries is invited to jump directly to Chapters 5 and 6 on graph fibrations.

4.1 Automorphisms and the symmetry group of a graph

Historically, the concept of a group traces back to Joseph-Louis Lagrange (1766–1769), Paolo Ruffini (1765–1822), Niels Henrik Abel (1802–1829), and especially Évariste Galois (1811–1832), who developed it in connection with solutions of polynomial equations. It was further developed by Camille Jordan (1838–1922) and others, and rapidly spread to all main areas of mathematics, a development heavily influenced by Emmy Noether (1882–1935). The concepts of groupoids and fibrations are of more recent vintage: groupoids were introduced by Brandt (1927), and fibrations by Grothendieck (1959). Their main uses have been in algebraic geometry and algebraic topology.

Consider a dynamical system (system of ODEs)

$$\dot{x} = F(x) \quad (4.1)$$

which in coordinates becomes

$$\begin{aligned} \dot{x}_1 &= f_1(x_1, \dots, x_N) \\ \dot{x}_2 &= f_2(x_1, \dots, x_N) \\ &\vdots \\ \dot{x}_N &= f_N(x_1, \dots, x_N) \end{aligned} \quad (4.2)$$

When the dynamical system (4.2) is admissible for a graph, it inherits the structural symmetries of the graph. These symmetries are particularly important because they define synchronized clusters that do not depend on the details of the dynamics, i.e., on the form of the inputs $f_i(x_1, \dots, x_N)$.

Symmetries of networks have traditionally been described using group theory. Recall that the symmetry group of a network is the set of permutations of nodes preserving the adjacency structure of the network (Bollobás, 2012). A permutation with this property is called an automorphism. Any two automorphisms compose to give an automorphism, a key feature of the group concept.

Suppose that a permutation maps node i to node j in (4.1). If this permutation does not change the dynamical equations, which should be the case if it is an automorphism, then x_i and x_j should play the same role in the dynamical system, evolve in the same way, and thus be able to synchronize. On the graph, an interchange of variables corresponds to an interchange of nodes, and symmetries of the dynamical system (4.1) correspond to symmetries of the graph, that is, automorphisms. An automorphism implies that many features of the network are invariant when its nodes are interchanged. An example of such a feature is adjacency of nodes, encoded in the adjacency matrix $A = (a_{ij})$. In this case, a symmetry is a permutation of nodes that does not change the global connectivity of the network (Pecora et al., 2014; Abrams et al., 2016; Bollobás, 2012; Golubitsky and Stewart, 2003): if any two nodes are connected before the permutation, they stay connected after the permutation.

Of course, features that refer to specific nodes, such as ‘node 1 connects to node 2’, may not be invariant. The node labels must be permuted according to the automorphism to preserve connectivity.

4.1.1 Formal definitions

We now make these statements precise and give examples.

Definition 4.1 Permutation of a graph. A *permutation* π of a graph $G(V, E)$ is a bijective map from the set of nodes $V = \{1, \dots, N\}$ to itself, $\pi : V \rightarrow V$. It permutes the node labels.

For example, the permutation π shown in the graph of Fig. 4.1 maps 1 to 1, 2 to 3, 3 to 2, 4 to 5, and 5 to 4. It is denoted by:

$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & 3 & 2 & 5 & 4 \end{pmatrix}, \quad (4.3)$$

in Cauchy’s two-line notation which consists of a first row with the original labels of nodes, and a second row with the permuted labels. Sometimes, for clarity, vertical arrows are inserted between the two rows. A compact alternative is cycle notation: $\pi = (2\ 3)(4\ 5)$, meaning that $2 \rightarrow 3 \rightarrow 2$ and $4 \rightarrow 5 \rightarrow 4$.

A *permutation matrix* P is an $N \times N$ matrix that is obtained from the identity matrix by

permuting both rows and columns according to π . In the case of permutation (4.3), this is:

$$P = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}. \quad (4.4)$$

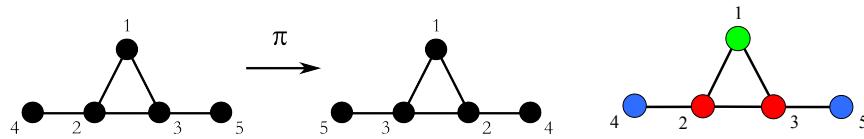


Fig. 4.1

Symmetric permutation. *Left:* Permutation (4.3) is an example of symmetric permutation preserving adjacency of nodes. *Right:* Corresponding orbital partition. Nodes that are interchanged are assigned the same color.

Definition 4.2 Automorphism (group symmetry) of a graph (Harary, 1993). An *automorphism* or *group symmetry* of a graph $G = (V, E)$ is a permutation $\pi : V \rightarrow V$ of the vertex set V , such that the pair of vertices i and j forms an edge (i, j) if and only if $(\pi(i), \pi(j))$ also forms a edge.

That is, an automorphism preserves adjacency and non-adjacency in the graph; two edges are adjacent after the permutation if and only if they were adjacent before the permutation. The set of automorphisms of a graph with adjacency matrix A , with the operation of composition, forms the automorphism group, or symmetry group, of the graph. We use the following notation:

Definition 4.3 Symmetry/automorphism group of a graph. Given a graph $G(V, E)$, we define:

$$\text{Aut}(G) = \{\pi \mid \pi \text{ is a symmetry permutation}\} \quad (4.5)$$

to be the set of all symmetry permutations. Composition preserves the network structure, so the set $\text{Aut}(G)$ is a group under composition and defines the *symmetry group* or *automorphism group* of the graph. It is a subgroup of \mathbb{S}_N .

An automorphism of a graph is a permutation of nodes that preserves the adjacency matrix (Babai, 1996). Any permutation matrix of a permutation π transforms the adjacency matrix into another A' as $A' = PAP^{-1}$. If P represents an automorphism, then $A' = A$, so

$$A = PAP^{-1}. \quad (4.6)$$

Since the group consist of matrices we can state this condition in another way. Equation (4.6) holds if and only if the matrix P commutes with A , i.e., $PA = AP$. Equivalently, the *commutator* is zero:

$$[P, A] = PA - AP = \mathbf{0}. \quad (4.7)$$

The permutation (4.3) is a symmetry of the network in Fig. 4.1. For a network with N nodes there are $N!$ possible permutations forming the symmetric group.

Definition 4.4 Symmetric group. The set of all permutations of the nodes of a graph $G(V, E)$ forms a group $\mathbb{S}_N = \{P_1, \dots, P_K\}$ where $K = N!$. It is called the *symmetric group* (not to be confused with a symmetry group).

Some of these permutations are symmetries, and the rest are not. The example network in Fig. 4.1 has $5! = 120$ permutations, 2 of which are symmetries (π defined in (4.3) and the identity) and 118 are not.

The symmetry group of a network can be calculated from the adjacency matrix A using widely available discrete algebra routines (Stein *et al.*, 2006; Linton, 2007). The problem of finding all automorphisms of a graph (or the generators of the group) is known as the Graph Automorphism Problem. Although in the worst case there is no polynomial-time algorithm to solve this problem, several algorithms work quite well for large graphs in practical applications. A popular algorithm to find automorphisms of a graph is McKay's *Nauty* algorithm (McKay, 1990), which is based on the well-known problem of testing isomorphism between graphs (see Section 5.7). Other algorithms based on Nauty are *bliss* (Junttila and Kaski, 2007) and *saucy* (Darga *et al.*, 2008), which is particularly good for sparse graphs.

The *identity permutation*, denoted by *id*, is the transformation that does not move any nodes in the graph. The identity is called a *trivial* automorphism. All graphs have a symmetry group that is composed at least of the identity. We will find that in most real biological graphs, the identity is the only automorphism of the graph.

Definition 4.5 A graph with no nontrivial automorphisms has only the identity permutation as a symmetry. Such a graph is said to be *rigid*. We also refer to this kind of graph as a graph with ‘no’ group symmetries (‘nontrivial’ is understood here: the identity map is always a symmetry).

However, we repeat that in this book the word ‘symmetry’ has a more general meaning than ‘automorphism’: it also refers to fibration symmetry. Therefore we do not call a rigid graph an *asymmetric* graph, as is common in the mathematical literature.

4.2 Cluster synchronization by automorphisms: orbital partition

Next, we discuss *orbital partitions* arising from automorphisms of graphs. This idea is already present in (Golubitsky and Stewart, 1985), applied to Hopf bifurcation for rings of oscillators, and is a special case of results in (Golubitsky *et al.*, 1988) for general symmetric dynamical systems. Pecora, Sorrentino, and others studied orbital partitions and cluster synchronization employing group symmetries and automorphisms in a series of papers (Pecora *et al.*, 2014; Sorrentino *et al.*, 2016; Klickstein *et al.*, 2019; Nishikawa and Motter,

2016; Schaub et al., 2016). The connection between symmetry and synchronization in networks with group symmetry is explained in detail in (Golubitsky and Stewart, 2023, Chapter 16). Other authors have studied group symmetries in real networks (MacArthur et al., 2008; Sánchez-García, 2020) as well as in multilayer graphs Della Rossa et al. (2020) and hypergraphs (Mulas et al., 2020).

Group symmetries, either by the whole automorphism group of a subgroup, naturally induce certain kinds of synchrony clusters. The set of automorphisms of the graph permutes certain subsets of nodes among each other. When the symmetry group $\text{Aut}(G)$ acts on the network, a given node i is moved by the permutations of the group to various other nodes j . In the language of group theory, the set of all nodes to which i can be moved defines the *orbit* of node i . Two nodes i and j are in the same orbit if there is an automorphism that maps i to j . This in turn defines the orbital partition of the network:

Definition 4.6 Orbit of a node and orbital partition. Given a graph $G(V, E)$, the *orbit* of a node $i \in V$ for a subgroup H of $\text{Aut}(G)$ is:

$$\mathcal{S}_H(i) = \{j \in V \mid \exists \pi \in H \text{ s.t. } \pi(i) = j\}. \quad (4.8)$$

If $H = \text{Aut}(G)$ we write $\mathcal{S}(i)$ rather than $\mathcal{S}_H(i)$. It can easily be proved that two orbits $\mathcal{S}_H(i)$ and $\mathcal{S}_H(j)$ are either equal or disjoint, and the union of all orbits is equal to V . Therefore the set of all orbits induces a partition of the nodes into mutually disjoint clusters. This set of all orbits forms the *H-orbital partition*. When $H = \text{Aut}(G)$ we call this the *orbital partition*: it is the *H-orbital partition* into the fewest clusters.

For examples, in the network of Fig. 4.1 the orbits are $\{1\}$, colored green; $\{2, 3\}$, colored red; and $\{4, 5\}$, colored blue.

There is a similar concept of an *H*-orbital partition for a subgroup H of the automorphism group.

4.2.1 Orbits define clusters

The orbits of subgroups of the symmetry group determine clusters of nodes that can synchronize. In other words, these orbits guarantee that the synchrony subspace of the cluster is flow-invariant (Golubitsky et al., 1988, Lemma XIII.2.1), (Pecora et al., 2014).

Definition 4.7 Cluster synchronization in orbital partitions. All nodes in the orbit $\mathcal{S}_H(i)$ of node i have the same dynamical state as node i , i.e.,

$$x_i(t) = x_j(t) \text{ if } j \in \mathcal{S}_H(i), \quad (4.9)$$

forming a synchronization cluster of an admissible system of equations to the graph.

A different question is whether the corresponding synchronous solution is stable; stability here is referred to as *synchronizability* and treated in detail in chapter 8. A third question is whether any particular cluster pattern occurs for a specific admissible differential equation, stably or not.

4.2.2 Orbital clusters and synchrony

To exemplify the connection between the orbits of the symmetry group $\text{Aut}(G)$ and cluster synchronization, we use the network in Fig. 4.1. A simple example of an admissible dynamical system compatible with this network (not the most general, see the Remark below) is of the form

$$\dot{x}_i = f(x_i) + \sum_{j=1}^5 A_{ij}g(x_i, x_j), \quad (4.10)$$

or explicitly

$$\begin{aligned}\dot{x}_1 &= f(x_1) + g(x_1, x_2) + g(x_1, x_3), \\ \dot{x}_2 &= f(x_2) + g(x_2, x_1) + g(x_2, x_3) + g(x_2, x_4), \\ \dot{x}_3 &= f(x_3) + g(x_3, x_1) + g(x_3, x_2) + g(x_3, x_5), \\ \dot{x}_4 &= f(x_4) + g(x_4, x_2), \\ \dot{x}_5 &= f(x_5) + g(x_5, x_3).\end{aligned}\quad (4.11)$$

Notice that we use the same function $g(\cdot)$ in all equations. This makes the system admissible for the graph, since the graph has all edges of the same type.

Remark In this example, specific functions have been assigned to edges, and interactions are assumed to be additive. These assumptions are common in biological models, and are often appropriate, but they are special cases.

In the general formalism for network dynamics used in this book, couplings need not be additive. In the example above, the most general admissible system has the form

$$\begin{aligned}\dot{x}_1 &= f(x_1, \overline{x_2, x_3}), \\ \dot{x}_2 &= g(x_2, \overline{x_1, x_3, x_4}), \\ \dot{x}_3 &= g(x_3, \overline{x_1, x_2, x_5}), \\ \dot{x}_4 &= h(x_4, x_2), \\ \dot{x}_5 &= h(x_5, x_3).\end{aligned}\quad (4.12)$$

for three distinct and arbitrary functions f, g, h with appropriate domains and ranges. Here the overlines indicate that the functions are symmetric under all permutations of those variables.

This level of generality is required in the theory of admissible ODEs because many models do not assign coupling functions in an additive (or multiplicative) manner.

The symmetry group of this network is $\text{Aut}(G) = \{\pi, id\}$, where $\pi = (2\ 3)(4\ 5)$, as in (4.3), and id is the identity. The orbits of the network are obtained as follows. We take node 2 and apply π to get node 3 and *vice versa*. This defines the first orbit $\mathcal{S}_1 = \{2, 3\}$. We repeat with node 4 and obtain $\mathcal{S}_2 = \{4, 5\}$. Node 1 just leads to a trivial cluster $\{1\}$.

We now show that nodes in these orbits synchronize. Applying π as in (4.3) to both sides

of (4.11) we get the transformed dynamical system:

$$\begin{aligned}\dot{x}_1 &= f(x_1) + g(x_1, x_3) + g(x_1, x_2), \\ \dot{x}_3 &= f(x_3) + g(x_3, x_1) + g(x_3, x_2) + g(x_3, x_5), \\ \dot{x}_2 &= f(x_2) + g(x_2, x_1) + g(x_2, x_3) + g(x_2, x_4), \\ \dot{x}_5 &= f(x_5) + g(x_5, x_3), \\ \dot{x}_4 &= f(x_4) + g(x_4, x_2),\end{aligned}\tag{4.13}$$

which is identical to the original dynamical system (4.11) before the permutation (though written in a different order). The invariance of (4.11) under π implies that if $x = (x_1, x_2, x_3, x_4, x_5)$ is a solution to (4.11), then $\pi(x) = (x_1, x_3, x_2, x_5, x_4)$ is also a solution, since it solves the same equations.

Moreover, the fixed points of the automorphism, given by $\pi(x) = x$, are obtained by substituting $x_2(t) = x_3(t)$ and $x_4(t) = x_5(t)$ in the equations, and the resulting system is consistent. Indeed, the ODE reduces to just three components, one for each cluster:

$$\begin{aligned}\dot{x}_1 &= f(x_1, \overline{x_2, x_2}), \\ \dot{x}_2 &= g(x_2, \overline{x_1, x_2, x_4}), \\ \dot{x}_4 &= h(x_4, x_2).\end{aligned}\tag{4.14}$$

After the substitution, the equation for \dot{x}_3 is the same as that for \dot{x}_2 , and the equation for \dot{x}_5 is the same as that for \dot{x}_4 .

Definition 4.8 Minimal orbital partition. For a given network there may be several orbital partitions. Of all the orbital partitions, there is exactly one partition with the smallest number of orbits. We refer to this as the *minimal orbital partition*.

Unless the symmetry group is trivial or has few subgroups, there are also non-minimal orbital partitions. These are generated by applying Definition 4.6 with a given subgroup H of the symmetry group. Furthermore, as we discuss in Chapter 5, there are other partitions that lead to cluster synchronization without automorphisms, and generally a graph may have many more symmetries than those captured by the orbits. These symmetries correspond to more general partitions, called ‘equitable’; equivalently to balanced colorings, which can also be viewed as fibers, which respect fibration symmetries.

To summarize, the orbits define synchronous clusters, but there can be other synchronized clusters that are not orbits. Orbital partitions are equitable, but not all equitable partitions are orbits (more on this in section 5.8). Permutation symmetries form a symmetry group with the same algebraic structure as groups in physics, including those groups associated with quantum mechanics and particle physics (Landau and Lifshitz, 1977; Weinberg, 1995).

While symmetry groups are all-pervasive in physics, permutation symmetry groups are rare in biological networks. Instead, the more general symmetry of fibrations captures biological symmetry. Among the few examples of biological networks with nontrivial automorphisms are the forward and backward locomotion networks of the nematode *C. elegans*, which are subgraphs of the neural network which contains only 302 neurons

and has been fully mapped: see section 23.3. In general, automorphisms seldom occur in biological networks, but fibrations are common.

4.2.3 Two types of quotient graph

The graph-theoretic literature includes a construction for the quotient of a graph by its automorphism group—or more generally by any group of automorphisms; see (Hahn and Tardif, 1997). This construction is natural in that context. The nodes of this quotient graph correspond to the orbits of the automorphism group (or subgroup), and therefore determine a balanced coloring, the ‘orbit coloring’ (Golubitsky and Stewart, 2023, Section 16.6). The dynamics of this synchronous state can be found from the model ODE concerned.

However, to avoid confusion, we emphasize that it is not the most appropriate notion of a quotient to represent synchronous dynamics. The reason is that although the nodes represent the clusters, the edges of this quotient graph do not represent the couplings inherited from the original graph in a manner that preserves the dynamics of all admissible ODEs. We expand on this point below in Example 2.

Suppose that G is a graph (directed or undirected) with automorphism group $\text{Aut}(G)$. There is a balanced coloring in which all nodes in an orbit of $\text{Aut}(G)$ are assigned the same color. The quotient as defined in (Hahn and Tardif, 1997) collapses the nodes in each orbit to form a single node, and defines edges between pairs of orbits in terms of edges between pairs of nodes, one in each orbit:

Definition 4.9 Quotient graph by an automorphism group in the sense of Hahn and Tardif (1997). Let G be a graph and let $\mathcal{S}(i)$ be the orbits of the automorphism group $\text{Aut}(G)$ of the graph. The *quotient graph* of the symmetry group acting on graph G is a graph whose vertices are the orbits $\mathcal{S}(i)$:

$$G/\text{Aut}(G) = \{\mathcal{S}(i) : i \in G\} \quad (4.15)$$

with a unique edge $\mathcal{S}(j) \rightarrow \mathcal{S}(i)$ whenever some (hence any) node in $\mathcal{S}(j)$ is connected to some node in $\mathcal{S}(i)$.

The mapping $\pi_{\mathcal{P}} : V(G) \rightarrow V(G/\mathcal{P})$ defined by $\pi_{\mathcal{P}}(u) = V_i$ such that $u \in V_i$, is the *natural map* for \mathcal{P} .

Although this notion of a quotient graph is well-known, it is not entirely appropriate for our purposes. The reason is that nodes in $G/\text{Aut}(G)$ are connected either by a unique edge, or none. This implies that in general the natural map is not a fibration: it fails to preserve input sets or input trees. Therefore the admissible ODEs for this notion of a quotient graph are not the same as the restrictions of the admissible ODEs for G to the synchrony subspace that corresponds to the clusters.

The next example illustrates this point and clarifies what we mean.

Example 2 The admissible ODEs for the graph G of Fig. 4.2a are:

$$\begin{aligned}\dot{x}_1 &= f(x_1, x_6) \\ \dot{x}_2 &= f(x_2, x_1) \\ \dot{x}_3 &= g(x_3, x_2, x_5) \\ \dot{x}_4 &= f(x_4, x_3) \\ \dot{x}_5 &= f(x_5, x_4) \\ \dot{x}_6 &= g(x_6, x_2, x_5)\end{aligned}\tag{4.16}$$

where g is symmetric in the second and third variables.

When restricted to the synchrony subspace corresponding to the orbits of $\text{Aut}(G)$, equation 4.16 takes the form

$$\begin{aligned}\dot{x}_1 &= f(x_1, x_3) \\ \dot{x}_2 &= f(x_2, x_1) \\ \dot{x}_3 &= g(x_3, x_2, x_2)\end{aligned}\tag{4.17}$$

This is the general admissible ODE for Fig. 4.2c, because of the double arrow. This network is the quotient network of 4.2a determined by the coloring, in the sense we use in this book: see Definition 6.9 for a general construction.

In contrast, the general admissible ODE for $G/\text{Aut}(G)$, namely Fig. 4.2b, is:

$$\begin{aligned}\dot{x}_1 &= f(x_1, x_3) \\ \dot{x}_2 &= f(x_2, x_1) \\ \dot{x}_3 &= f(x_3, x_2)\end{aligned}\tag{4.18}$$

Equation 4.18 differs from 4.17. In particular, 4.18 involves a single function f and has a \mathbb{Z}_3 automorphism group. This leads to generic dynamics that would not be generic for 4.17. The latter involves a second function g , and has trivial automorphism group. Its dynamics reduces to 4.18 if we choose g so that $g(u, v, w) = f(u, v)$, so the quotient network in the sense of this book (Definition 6.9) has more general dynamics.

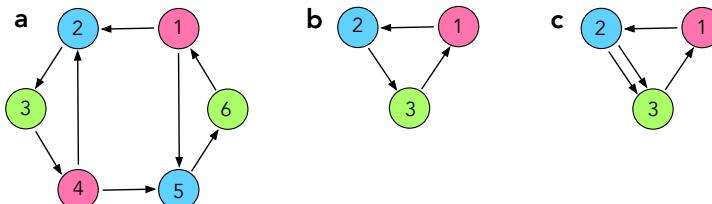


Fig. 4.2

Quotient graph. Difference between the quotient graph as defined in (Hahn and Tardif, 1997) and the quotient network for the corresponding fibration. (a) A directed graph with six nodes and automorphism group \mathbb{Z}_2 generated by 180° rotation $(1\ 4)(2\ 5)(3\ 6)$. The coloring is the orbital coloring. (b) Quotient graph in the sense of (Hahn and Tardif, 1997). This is not the base of a fibration because the input arrow to node 2 does not lift uniquely, see chapter 6. (c) Quotient network defined by the fibration corresponding to the colors has an extra arrow from node 2 to node 3, and is (necessarily) a multigraph.

Input Trees, Synchrony, and Equitable Partitions

Robust synchrony patterns in networks correspond precisely to fibration symmetries, or equivalently equitable partitions of the nodes, also called balanced colorings. We motivate these notions using two simple biological networks, the metabolator and the Smolen oscillator. Each has two nodes, which can synchronize. For the metabolator, the explanation is symmetry in the group-theoretic sense. For the Smolen oscillator, however, the symmetry group is trivial, and the explanation of synchrony is a fibration. We discuss the key difference between global group-theoretic symmetries and local fibration symmetries. We state a series of modeling assumptions that can naturally create synchrony, and show how they lead to the fibration concept. A crucial mathematical point is that fibrations are determined by the structure of the input trees of nodes, which represent the flow of information through the network. The minimal equitable partition (largest clusters of synchronous nodes) is determined by isomorphism of input trees.

5.1 Intuition and motivation

Even though automorphisms of a graph impose a rigid type of symmetry with strict conditions on the graph structure, it seems to be widely believed that these symmetries determine all possible synchrony patterns in a network. Simple examples show that this is false. We already saw this for the UxuR GRN in Fig. 2.1. We now compare two even simpler networks, which we revisit to illustrate fibrations in Chapter 6. Each has two nodes; one is symmetric under interchange of the nodes, and the other is not. Figure 5.1 shows these networks, the metabolator on the left and the Smolen oscillator on the right, which occur in synthetic genetic oscillators (Purcell et al., 2010). Here, two genes regulate each other, and also self-regulate themselves as activators (pointed) or repressors (bar).

Both of these networks support synchronous states, where both nodes have identical states. For the metabolator, this is explained by its \mathbb{Z}_2 group symmetry, which permutes the two nodes. Admissible ODEs have the form

$$\dot{x}_1 = f(x_1, x_2) \quad (5.1)$$

$$\dot{x}_2 = f(x_2, x_1) \quad (5.2)$$

which both reduce to $\dot{x} = f(x, x)$ when $x_1 = x_2 = x$. So this equation describes the synchronous states. However, the Smolen oscillator has trivial symmetry group (only the identity), yet it also supports the same synchronous state. Now admissible ODEs have the



Fig. 5.1

Synchronization without group symmetry. *Left:* Metabolator network. *Right:* Smolen oscillator network. Arrowheads indicate activator (pointed) and repressor (bar). In both graphs, nodes 1 and 2 are found to synchronize. In the metabolator, this synchronization can be explained by the group symmetry of the graph (and also by the fibration symmetry since it includes automorphisms). In the Smolen oscillator the synchronization is not explained by the group symmetry since this graph has no nontrivial automorphisms. Instead, the fibration symmetry captures the synchronization.

form

$$\dot{x}_1 = f(x_1, x_2) \quad (5.3)$$

$$\dot{x}_2 = f(x_1, x_2) \quad (5.4)$$

and again, both reduce to $\dot{x} = f(x, x)$ when $x_1 = x_2 = x$. That is, node 1 synchronizes with node 2 in both graphs. How can we explain this synchronization in the absence of nontrivial group symmetry?

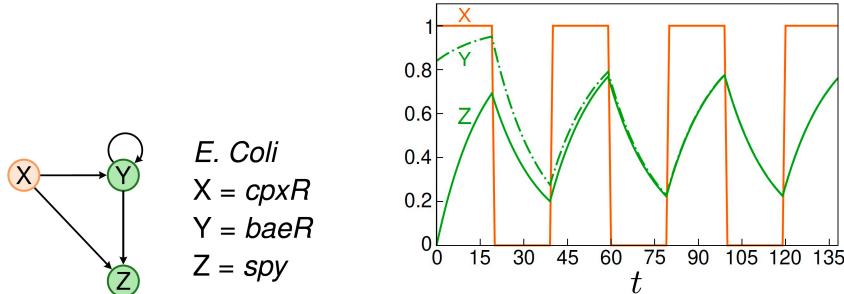


Fig. 5.2

Synchronization without a symmetry group in a gene regulatory circuit. As an example, we consider a building block of the gene regulatory network of *E. coli* called the feed-forward fiber (Leifer et al., 2020), composed of genes $cpxR = X$, $baeR = Y$, and $spy = Z$. The circuit has no group symmetry, yet a numerical solution of the dynamics shows synchronization of the expression levels of genes Y and Z.

We can further illustrate this conundrum with the circuit depicted in Fig. 5.2 left, which is called a feed-forward fiber (FFF). This circuit is a fundamental fibration building block of gene regulatory networks. It is found abundantly in *E. coli* as well as in other biological networks (Morone et al., 2020; Leifer et al., 2020), and will be discussed in detail in Chapter 15. It is similar to the UxuR circuit of Fig. 2.1, but now all edges have the same type. The FFF circuit consists of a feed-forward loop motif as found in (Milo et al., 2002) between genes X, Y and Z, plus an autoregulation loop at Y. Again, this circuit has no nontrivial group symmetries. However, numerical simulations shown in Fig. 5.2 right and analytical

solutions by Leifer et al. (2020) confirm the existence of cluster synchronization of genes Y and Z. We use the model equations

$$\begin{aligned} y_{t+1} - y_t &= -\alpha y_t + \gamma_x \theta(x_t - k_x) \gamma_y \theta(y_t - k_y) \\ z_{t+1} - z_t &= -\alpha z_t + \gamma_x \theta(x_t - k_x) \gamma_y \theta(y_t - k_y) \end{aligned}$$

with parameters $\alpha = 0.06$, $\gamma_x = 0.775$, $\gamma_y = 0.775$, $k_x = 0.5$, $k_y = 0.1$, $y_0 = 0.85$, $z_0 = 0.0$. Notice that the chosen parameters make this model system admissible for the graph.

5.2 Global symmetries versus local symmetries

Fibrations are more relevant to biology than automorphisms because they are more robust, often surviving changes to the network. Automorphisms are far more delicate. To see why, we compare fibrations and automorphisms using the graph in Fig. 1.3, which is replotted in Fig. 5.3b together with a slight variation in Fig. 5.3a. The graph of Fig. 5.3a exhibits a global left-right group symmetry defined by the automorphism:

$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 3 & 2 & 5 & 4 & 6 \end{pmatrix}, \quad (5.5)$$

or in cycle notation $\pi = (2\ 3)(4\ 5)$; nodes are connected in exactly the same way before and after applying π . This automorphism preserves the connectivity of the entire graph and represents a global symmetry of the network. In particular, it preserves both the inputs and the outputs of the nodes, as specified by the adjacency matrix. This automorphism leads to a partition into four orbits, as defined in Definition 4.6; the nontrivial ones are composed of nodes $S(2) = \{2, 3\}$ and $S(4) = \{4, 5\}$. Nodes in these orbits can synchronize in clusters. *Both* orbits must synchronize simultaneously to create such a state, which satisfies

$$x_2 = x_3 \quad x_4 = x_5.$$

Next, consider the slightly modified graph depicted on the left of Fig. 5.3b, which differs from the network in Fig. 5.3a by one extra out-going edge from node 3 to 7. In this graph, the permutation of nodes $\pi = (2, 3)(4, 5)$ in (5.5) is no longer an automorphism, because it does not preserve the in and out connectivities of nodes 2, 3 and 7. For instance, node 3 is connected to 7 in the original graph but it is not after the permutation is applied (Fig. 5.3b right). Likewise, the connectivity of nodes 2 and 7 is different before and after the permutation. There are no other symmetries in this graph.

We see how fragile group symmetries are: by adding just one outgoing link from the symmetric graph (the edge $3 \rightarrow 7$), the global symmetry is broken. The automorphism group of the network in Fig. 5.3b now contains only the trivial identity permutation. This fragility occurs because automorphisms require very strict global arrangements of nodes and links to preserve the global structure of the network.

The breaking of global symmetry by adding edge $3 \rightarrow 7$ changes the automorphism group of the graph, and also changes its orbits. The predicted absence of an orbital partition and concomitant cluster synchronization is significant, because it represents a global

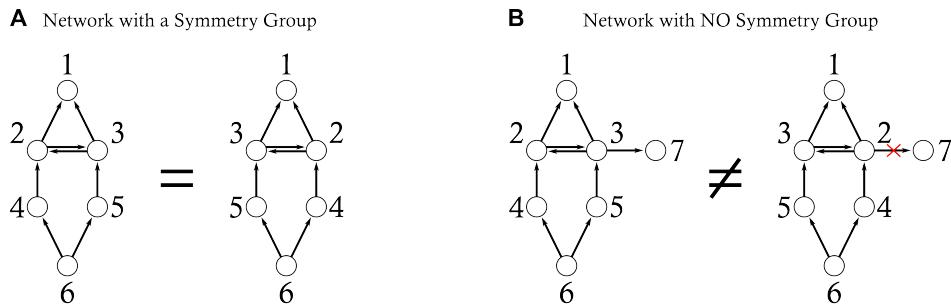


Fig. 5.3

Group symmetries and fibrations. (a) Example of a network with a (nontrivial) symmetry group. The automorphism shown maps the network to itself leaving invariant the connectivity of every node in the network. (b) A graph without (nontrivial) automorphisms but with a fibration symmetry. The addition of a single out-going edge from $3 \rightarrow 7$ breaks the global group symmetry of the graph rendering the graph with only a trivial group symmetry (the identity). We will see that, even though this graph has no group symmetry, it is still symmetric under a symmetry fibration: see Fig. 5.14.

vulnerability to the perturbation. That is, the addition of outgoing link $3 \rightarrow 7$, which in principle should not affect the dynamics of node 3 and much less that of nodes 4 or 5, breaks the synchronization of not only the orbit $S(2) = \{3, 4\}$ but also of the other orbit $S(4) = \{4, 5\}$, even though the added link does not directly connect to this orbit nor emanate from it. This exemplifies the nonlocal vulnerability of group symmetry. It also shows that (group) symmetry breaking is different from synchrony-breaking.

5.2.1 Biology is robust

This example shows that an orbital partition based on a global symmetry cannot properly capture cluster synchronization: a change in any part of the network can produce a loss of synchronization in an orbit located in another part of the network that is not even connected with the site of this change. Such a vulnerable biological network would not survive under evolutionary pressure. A resilient evolving network requires a more robust type of symmetry that preserves local synchronization under many small changes to the network, or the equations modeling its dynamics. Such a local symmetry could guarantee robustness of the synchronization, while at the same time providing flexibility to changes in the network structure for evolvability of new phenotypes under natural selection.

This example raises the following question: although there are no (nontrivial) automorphisms, are there extra symmetries in the network shown in Fig. 5.3b that could lead to robust synchronization? The answer to this question is, ‘yes’ in this case: there are fibration symmetries. Fibration symmetries, by not being global symmetries of the network, allow more robust patterns of synchrony under random addition or deletion of nodes. This fact allows the network to evolve more freely than one that is constrained by symmetry groups would do, and can still preserve synchronization.

The same example shows that synchronization is usually not a property of a single cluster. Instead, it is a collective property of a set of clusters, corresponding to the partition (coloring) determined by the clusters. So when we say that certain nodes ‘can synchronize’, this statement is always made in the context of a set of clusters, and usually requires other sets of nodes to synchronize as well.

Automorphisms always preserve outputs as well as inputs. Fibrations preserve only inputs. This is one reason why they are more common. In undirected graphs every input is also an output, so fibrations are likely to be more common in directed networks. Most biological networks are directed. That said, most directed networks, constructed randomly, have only the trivial fibration symmetry. Moreover, reversing the extra arrow $2 \rightarrow 7$ in Fig. 5.3b to get $7 \rightarrow 2$ also destroys the group symmetry, so the input/output distinction is not the only reason for the loss of symmetry.

In summary: automorphisms describe invariances in all physical systems (elementary particles, forces, atoms and molecules) but are rare in biological networks, perhaps due to their inherent vulnerabilities. Next, we will see how fibration symmetries—with their emphasis on preserving only input trees locally but not globally—offer a less constrained and more robust alternative to symmetry groups, still producing the necessary synchronization required for biological function.

5.3 Modeling assumptions leading to synchrony

As discussed in section 3.7, one of our aims is to decipher the structure \rightsquigarrow function relation in biology. This means understanding how the graph structure determines the dynamics of the biological system and, in turn, the biological functions of its components. Our purpose is to decouple the dynamics from the structure of the graph, making predictions of synchronization based on graph properties alone. The way to achieve this is via the fibration formalism. To make theoretical progress in this direction we start with a series of modeling assumptions about conditions that permit synchrony. In Chapter 6 we show how these assumptions let us apply fibrations to biological graphs. Later, we relax these assumptions by switching to more realistic models of biological networks, tested by experimental data on biological synchronization, but the same basic principles continue to hold.

We propose that processes in cellular networks can be seen as computations. Initially, for simplicity, we think of a biological network or, more generally, an information-processing network, as a Von Neumann cellular automaton or ‘deterministic finite-state machine’: a computational machine made of units passing signals or ‘messages’ through directed connections to perform some kind of computation based on their previous states (Leifer et al., 2020). In such models, the phase spaces of nodes are discrete (indeed, finite). More complex dynamics such as ODEs are analogous, and the same metaphors apply, subject to suitable interpretations.

These signals could represent neuronal spiking, forming the basis for information transfer in the brain, or ‘message passing’ from source gene to target gene in a transcriptional regulatory network, where a transcription factor expressed by the source gene diffuses in

the cell until it binds to the promoter DNA site of the target gene to turn it on and off. Our aim is to capture key aspects of the more complex reality in a simple model, and we start with the simplest setting.

In a directed network, nodes pass signals or messages from source to target along the directed links. In particular, nodes can receive signals from other nodes. This information flow respects the direction of edges: if there is an edge from node i to node j , that connection can be used to send information *from i to j*, but not the other way around. Based on the signals they receive, nodes compute and then broadcast further signals through their outgoing edges, and so on. The nodes of such a network are autonomous processing entities, behaving according to well-defined rules. More precisely, we start from the following three assumptions. Later we add two more.

- **States:** Every node, at every moment in time, finds itself in a certain state.
- **Determinism:** Its future behavior depends deterministically (i.e., in a uniquely prescribed way) only on its current state and on the inputs it receives from incoming signals.
- **Broadcast:** When a node sends out a signal, it sends the same signal through all of its outgoing edges of the same type.

While the first two assumptions are quite natural (and are the essence of a Von Neumann cellular automaton or deterministic finite-state machine), the last one is more puzzling. What we are saying is that, for example, when node 6 in the network of Fig. 5.3b communicates a signal, it will communicate the same piece of information both to node 4 and to node 5 (the only two nodes it can communicate to). This postulate is not as restrictive as it might seem, and can be circumvented using typed edges. For example, to enable node 6 to communicate differently with nodes 4 and 5, we give the edges $6 \rightarrow 4$ and $6 \rightarrow 5$ different types. For instance, one edge could be an activator and the other a repressor. Thus edge labels (often indicated in diagrams by different forms for the arrows) can capture different types of biological interactions, such as activators and repressors in a gene regulatory network, or excitatory/inhibitory connections in the brain. Thus the third constraint above can be relaxed even further by requiring the broadcast to be uniform only for the messages sent through a given type of edge. We will get back to this later, but for the moment, and for the sake of simplicity, we maintain the assumption above and refrain from using different types of edges.

Based on these assumptions, for instance, since node 6 of Fig. 5.3 has no incoming edges, it receives no signals, so its future behavior depends only on its initial state. In particular, this observation entails that the signals that node 6 sends to 4 and 5 depend only on its initial state, and nothing else: it can be considered as an external parameter of the system.

Now, consider nodes 5 and 4. They receive information only from node 6; moreover, they receive the same information because of the broadcast postulate. So if they start in the same state, they will remain in the same state because of the determinism postulate. Therefore they behave in synchrony, even though there is no graph automorphism that exchanges them.

In this chapter we are concerned only with showing that these synchronous states *exist*, as a consequence of fibration symmetries of the graph. However, we repeat that the existence of cluster synchronization does not guarantee that the solutions concerned are

dynamically stable. As already mentioned, the study of the stability and attractiveness of these synchronous states is referred to as synchronizability. The stability of the synchronized solution is a separate issue from its existence. A synchronous solution predicted by fibration symmetries or any other symmetry may result in an unstable state, which would not be biologically meaningful. We discuss stability in Chapter 8.

5.3.1 Necessary conditions for synchrony

To make one further step towards understanding the nature of biological networks, we continue to study the example of Fig. 5.3. We consider a subset S of nodes, and ask what conditions are required for S to be a synchronized cluster. We say that such a set can ‘potentially’ synchronize.

One obvious condition is that the concept of synchrony must make sense for the nodes concerned. This is not the case if, for example, one node has phase space \mathbb{R} (the real numbers) and the other has phase space S^1 (the circle, that is, angles), since it is not legitimate to compare a number with an angle.

In some types of model the number 0 plays a special role, indicating ‘no activity’, and such a comparison might be considered legitimate for some purposes, but the topology of the real line differs from that of the circle. In the theoretical parts of this book we do not assume that 0 is special; in particular we do not assume that the functions defining the dynamics in an admissible ODE vanish at their origin—an assumption that is often made in the literature. So, in general, there is no automatic complete synchronization state like those given by Laplacian dynamics in Section 3.6.1. Instead, the *structure of the network* is what makes synchronized states possible without building them into the model from the start.

Since nodes in a synchronized cluster are synchronized for all time, we avoid breaking synchrony via unsynchronized initial conditions by assuming that all nodes that potentially could synchronize are initially set to the same state. This condition assumes that ‘same state’ can meaningfully be defined for all nodes, which need not be the case in general, but it must make sense for nodes in a synchronized cluster, because it is the key feature defining synchrony. For this reason we add a fourth assumption:

- **Anonymity:** When seeking synchrony patterns, we may assume that all nodes that can potentially synchronize start from the same initial state.

Suppose we apply this reasoning to nodes 2 and 3. Node 2 receives signals from node 4 and from node 3, whereas node 3 receives signals from node 5 and from node 2. As we said, node 4 and node 5 behave in the same way, so node 2 and node 3 will receive the same signals along one of the edges (whenever some piece of information is sent along the edge $4 \rightarrow 2$, the same piece of information is sent along the edge $5 \rightarrow 3$). Nodes 2 and 3 also receive signals from each other, but again this cannot break their synchrony if they start in the same state.

The anonymity assumption might look very restrictive; indeed, no real biological system can satisfy it, since the initial states are constantly changing and the cell needs to adapt to continuously changing signals from the environment. Mathematically, it represents the

principle that any meaningful synchrony pattern should persist over time. Another way to say this is that the initial state lies in the appropriate synchrony subspace. In line with our purpose of decoupling the message-passing dynamics from the graph structure, this assumption is necessary at this stage. However, this restriction can easily be relaxed afterward. In particular, it does not imply that nodes that start in different states cannot tend toward synchrony as time passes. This condition, in fact, avoids the problem of synchronizability to be treated in Chapter 8.

To avoid ‘accidental’ synchrony patterns, it is convenient to add a fifth assumption:

- **No Fragility:** The synchrony pattern is not *fragile*, i.e., it does not depend on special features of the admissible equations. In particular, it is not destroyed by small admissible changes to the equations.

For example, consider Fig. 4.1. In an extreme case, suppose that the function g that represents the coupling is identically zero, so the equations decouple. Then, all five nodes obey the same equation $\dot{x}_i = f(x_i)$, so complete synchronization is possible. However, complete synchronization is *not* possible in this network with more general choices of g , because for many g the components \dot{x}_1 and \dot{x}_2 must satisfy different equations. This makes the equations logically inconsistent. In this case, what below we call the ‘minimal equitable partition’—the synchronization pattern with fewest clusters—is the pattern with clusters $\{1\}, \{2, 3\}, \{4, 5\}$.

Remark More seriously, the same problem arises if the coupling function g vanishes whenever the source and target nodes are synchronized; that is, if $g(y, y) = 0$ for all y . Again, the equations decouple and complete synchronization becomes possible. A common example of this kind of coupling is ‘diffusive coupling’. Since this is a common modeling assumption in some areas, the definition of a fibration or balanced coloring should be modified. This can be done (Golubitsky and Stewart, 2023, Section 8.9), see also Kamei and Cock (2013b); Díaz-Parra et al. (2017); Sorrentino et al. (2016); Siddique et al. (2018). Another widespread context in which coupling between synchronous nodes is always zero occurs in models based on the graph Laplacian. See Sections 5.9 and 8.10.

5.4 Input sets and input trees

We now study the consequences of the observations listed in the previous section, starting from the notions of input set and input tree. There are several definitions of the input set in the literature. We define it in a way that is most consistent with information-processing networks. Simply stated, the input set of a node is the set of incoming *edges*, and the number of them is the *in-degree* (or *valence*) of that node.

Using only edges may seem unnatural or pedantic, but it makes the mathematical formalism work better, and is necessary for multigraphs. Related information, such as source or target nodes of an edge, can be specified separately as additional structure. We also think of the node itself as a special type of edge, and include it in the input set. (In (Golubitsky

and Stewart, 2023) this version is called the ‘extended input set’. The node is included in the formalism as a distinguished variable in model equations.)

Definition 5.1 Input set of a node. The *input set* of the node $i \in V$ in a graph $G = (V, E)$ is the set $I_i = \{i\} \cup \{e \mid e \in E, t(e) = i\}$ that consists of the node itself and all of its incoming edges (which can be self-loops, if the node has any).

We denote the input set of a node $i \in V$ with in-degree k_{in} as $I_i = (i : e_1, \dots, e_{k_{\text{in}}})$, where $e_j \in E$ are the incoming edges. This also defines the *local in-neighborhood* of the node i in the terminology of graph fibrations (Boldi and Vigna, 2002a). An analogous notation also employed in the literature on symmetry groupoids is in terms of the nodes connecting to i from incoming edges, thus $I_i = (i : 1, \dots, k_{\text{in}})$, where $(1, \dots, k_{\text{in}}) \in V$ are the in-neighborhood nodes of i (see Fig. 5.5): here, we are assuming that the graph itself is simple.

Conceptually, the input set of i is the set of ‘objects’ that can influence i ’s behavior: it includes i itself (because the current state of the node will of course, influence its own future behavior) as well as all its incoming edges (because it is through those edges that input signals come, and input signals influence the node’s behavior). We deliberately prefer to use incoming edges instead of their source nodes because multigraphs can have several edges with the same source and the same target. This influences the behavior differently compared to a single edge (even when they all carry the same signal). However, it is the state of the source node that generates that signal: we do not equip edges with states.

That said, when drawing input sets, it is often convenient to include their source nodes as well as their target nodes. The source nodes are not part of the input set, but they add useful information for purposes such as the construction of admissible ODEs. In such drawings, distinct edges are drawn with distinct sources; however, if the nodes are numbered, some source nodes may correspond to the same number in the case of multigraphs. That is, the sources of multiple edges are split apart but numbered in a way that shows how to identify them.

An input set is naturally represented by a graph. Consider Fig. 5.4, where we show again the network of Fig. 5.3b left, but with names on the edges so that they can be identified; Fig. 5.5 shows the input sets of nodes 1, 2, 3 and 4.

Input sets can be put together inductively to form *input trees*, following (Morone et al., 2020). Input trees are referred to as an ‘infinite depth input tree’ in (Aldis, 2008; Stewart, 2007) and as a ‘universal total graph’ in (Boldi and Vigna, 2002a). Technically, the tree graph is a ‘universal cover’ of the original graph.

Definition 5.2 Input tree of a node. We inductively define the *input tree* T_i of a node $i \in G$ as follows:

- if $I_i = \{i\}$, then the input tree T_i is the root-only tree;
- if $I_i = (i : e_1, \dots, e_{k_{\text{in}}})$, then the input tree T_i is formed by a root with k_{in} incoming edges from its k_{in} input nodes, plus the subtrees rooted at the input nodes $T_{s(e_1)}, \dots, T_{s(e_{k_{\text{in}}})}$ in no particular order.

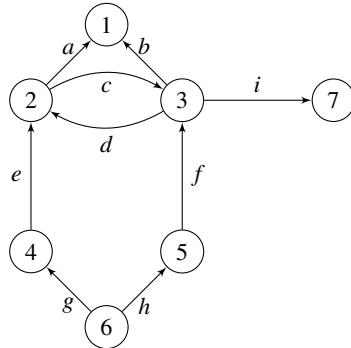


Fig. 5.4

A network with no automorphisms. The same as in Fig. 5.3b (left) but with labeled edges. Usually nodes are labeled with numbers and edges with letters.

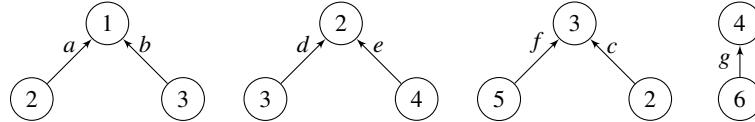


Fig. 5.5

Definition of input set. Input sets for the nodes 1, 2, 3 and 4 of the network of Fig. 5.4, together with their source and target nodes. We denote the input set of, for instance, node 1 as: $I_1 = (1 : a, b)$.

Again, only the edges are required, but it is convenient to add the target and sources to help specify paths through the graph.

Input trees can be finite or infinite, but they are infinite in most practical cases (as soon as there is a cycle in the graph). More precisely, T_i is infinite if and only if there is a node j that has a path to i and is involved in a cycle. (This cycle could be an ‘autoregulation’ loop from j to itself.)

As we said, input trees can be built by starting from input sets and stitching them together, and here the node labels are useful. In Fig. 5.6 we see the input tree T_2 for node 2 of the network of Fig. 5.4. Observe that the same node can (and usually does) occur repeatedly in different positions in the tree, and we do not identify these nodes in the tree diagram. So these node and edge labels are there only for clarity. An input tree can be imagined as the set of ways in which signals (or, if you prefer, states) can reach a given node. That is, the directed *paths* through the graph. For instance, node 2 (the root of the tree) can receive information from node 4 (through the edge named e), whose state can depend on the information it receives from node 6 (through the edge named g). This is equivalent to saying that the state of node 6 influences the state of node 4, which in turn influences the state of node 2. Paths through the tree correspond to paths through the original graph, but in the tree, distinct paths do not overlap or share common edges, as they may do in the graph.

Input trees can be better understood if we consider them as unfoldings of network behavior along the axis of time. This way of reasoning requires a further assumption:

- **Synchronous update:** all nodes change their state at the same time.

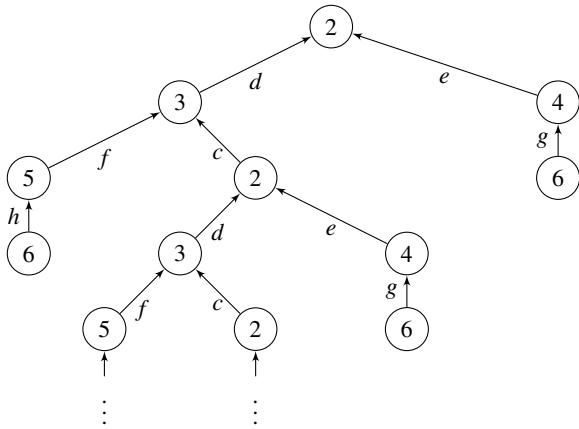


Fig. 5.6

Example of an input tree. The input tree T_2 of node 2 in the network of Fig. 5.4. The tree is infinite (we show only its upper levels).

This quite strong assumption can be thought of as yet another form of worst-case scenario: once more, we wish to look at the graph structure as the only source of asymmetry, so we keep all other sources of asymmetry (initial states of the node, communication time, broadcasting, etc) constant. These constraints can be relaxed later.

Remark Synchrony between nodes can be broken based on communication delays; that is, two otherwise symmetric nodes can receive the same signal with a different delay, which may cause asynchronous dynamics. In this case the edges of the network, which represent *couplings* between nodes, have different types, but a symmetry must preserve edge types. This chapter concerns the kind of symmetry breaking in which states of the system have less symmetry than that of the network. This is possible because nonlinear systems can support several distinct states simultaneously, selected by initial conditions, and a fully symmetric state may be unstable. Symmetry breaking caused by different delays, together with the question of the dynamics of networks with non-fixed architecture, are outside of the scope of this book.

Synchronous update implies that there is some global clock that keeps ticking, and at every tick, akin to a computer, simultaneously all nodes receive their incoming signals and change their state. This is ‘discrete’ dynamics. It is also possible to model the dynamics in continuous time using differential equations, and here the use of a common time variable for all nodes effectively updates their states synchronously. Again, there is a global clock, but it ‘ticks’ continuously.

Figure 5.7 shows the same input tree T_2 , but on a timeline: the axis of time is shown on the left, and nodes of the tree are aligned with time ticks. Thus, the state of node 2 at time 0 depends on the states of nodes 3 and 4 at times -1. In turn, the state of node 3 at time -1 depends on the states of nodes 5 and 2 at time -2 and so on.

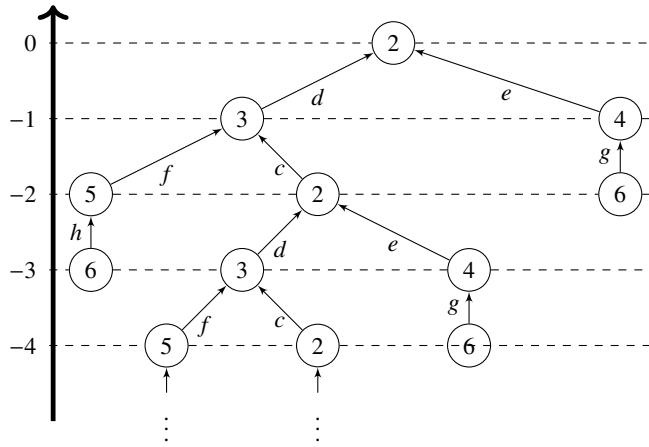


Fig. 5.7

Dynamical interpretation of the input tree. The input tree T_2 of node 2 in the network of Fig. 5.4, as a representation of the information flow represented as an unfolding in time of the signals that node 2 receives.

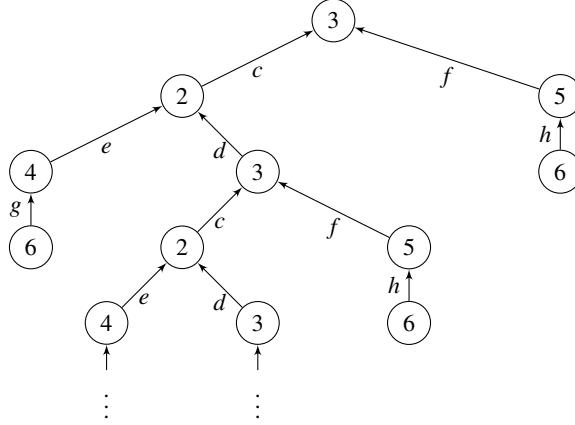


Fig. 5.8

Example of input tree. The input tree T_3 of node 3 in the network of Fig. 5.4. The tree is infinite (only its upper levels are shown).

Here we have assumed discrete dynamics. For continuous dynamics, the *whole tree* updates simultaneously.

The input tree of a node i can be thought of as the complete set of all pathways in the network that terminate at i (Morone et al., 2020; Boldi and Vigna, 2002a; Aldis, 2008) (the node itself can be a part of that path if it has a self-loop). Therefore, an input tree contains all the pathways that can contribute to the dynamics of the node, showing all the ‘information’ about the network that is available to the node, hence the only form of *asymmetry* that the node can use to differentiate itself from the other nodes in the network.

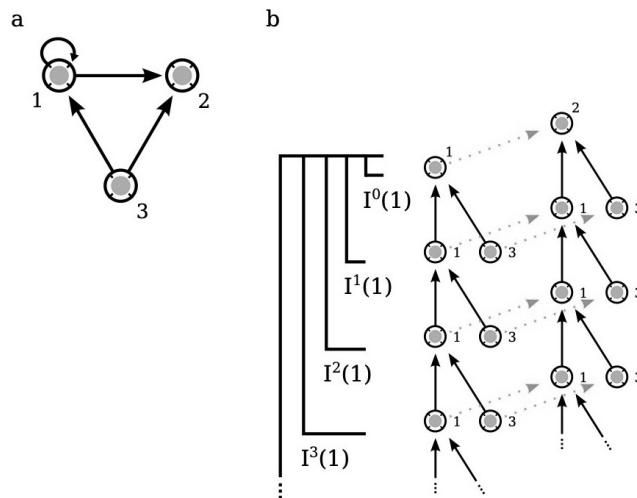


Fig. 5.9

Example of the input tree of the FFF. (a) The feed-forward fiber (FFF) circuit. (b) Input trees of the three nodes in the graph. The input set of node 1 consists of nodes 1 and 3 and the next layer of the input tree is the input set of 1 attached to node 1. The input set of node 3 is only the node itself. The procedure is repeated. The red dotted arrow shows the bijection between input trees I_1 and I_2 illustrating the definition 5.4 of graph isomorphism.

5.5 Examples of input trees

Input trees are defined inductively as input sets of input sets . . . taken an infinite number of times. We previewed one example in Fig. 2.2. We now discuss a biological example, shown in Fig. 5.9: the input tree of the FFF circuit, already discussed in Fig. 5.2. Figure 5.9 shows the process of constructing an infinite depth input tree for the FFF circuit. For instance, consider node 1. The root of the tree is the node itself. The first layer is the input set which contains nodes 3 and 1, since there is a self-loop at node 1 and 3 connects to 1. The next layer is constructed by attaching the input set of 1 to the leaf 1 and the input set of 3 to the leaf 3. The procedure is repeated *ad infinitum* to represent the full input tree. Even if the graph is finite, the input tree can be finite or infinite. If the graph has no cycles and self-loops, then all input trees are finite (Aldis, 2008). As soon as any node in the input tree belongs to a cycle (even just a self-loop), the input tree has an infinite number of layers. See also (Golubitsky and Stewart, 2023, Section 13.3).

5.6 Input tree isomorphism

Figure 5.6 and Fig. 5.8 show the input trees of node 2 and 3 (respectively) in the network of Fig. 5.4: although the node and edge labels appearing there are different, the shape is the same. A more precise statement would be that the two trees are isomorphic, which means ‘equal shape’. This is why nodes 2 and 3 can be exchanged by a fibration symmetry. We now formalize equivalence of the ‘shape’ of two graphs, while ignoring their labeling and the way they are drawn.

Definition 5.3 Graph isomorphism (Harary, 1969). Two graphs $G = \{N_G, E_G\}$ and $H = \{N_H, E_H\}$ are *isomorphic* if there exists a bijection $\alpha_N : N_G \rightarrow N_H$ between nodes and another bijection $\alpha_E : E_G \rightarrow E_H$ between edges, such that $s(\alpha_E(e)) = \alpha_N(s(e))$ and $t(\alpha_E(e)) = \alpha_N(t(e))$ for all $e \in E_G$. We denote isomorphism of G and H by

$$G \simeq H. \quad (5.6)$$

Usually we just write α instead of α_N and α_E , unless this would cause confusion. We say that α is a *graph isomorphism*.

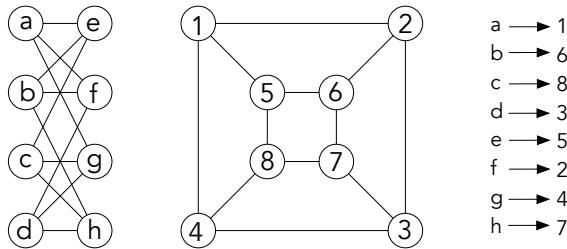


Fig. 5.10

Graph isomorphism. Two graphs G and H are isomorphic ($G \simeq H$) despite their different drawings and labels, meaning that when we relabel one of the graphs with the isomorphism shown on the right, the graphs are identical. Here labels are just numbers and letters, but in a biological network labels are genes (or proteins, neurons, and so on), and the graph isomorphism acquires an important dimension.

Simply speaking, an isomorphism α maps nodes and edges of G to nodes and edges of H in such a way that the image of the edge connects the image of the source of this edge to the image of the target, as in Figs. 5.10 and 5.11.

A graph automorphism is an isomorphism that maps a graph onto itself, i.e., $\alpha : G \rightarrow G$, see Fig. 5.11. An automorphism of a graph is a graph isomorphism with itself, i.e., a mapping from G back to G such that the resulting graph is isomorphic with G . The set of all automorphisms of G is a group under functional composition, and it is precisely the symmetry group described in Chapter 4.

In general, determining whether two graphs are isomorphic is a difficult problem. There

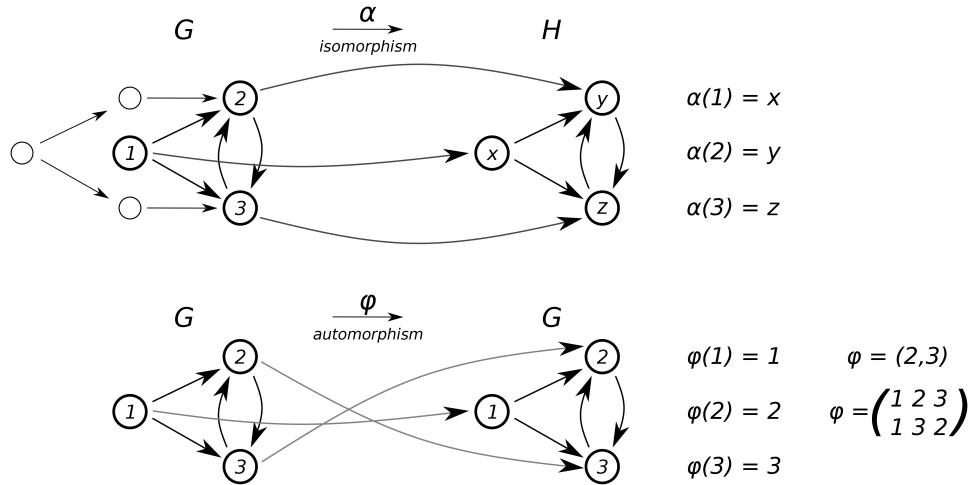


Fig. 5.11

Isomorphism and automorphism. Example of an isomorphism of a subgraph and an automorphism of the whole graph.

exists no known polynomial-time algorithm for graph isomorphism testing, although the problem has not been shown to be NP-complete. It is believed that the problem belongs to a low hierarchy of the class NP, but its exact standing is unknown and it is itself an intensely active field of research (Grohe and Schweitzer, 2020). However, we observe below that when the original graph is finite, there are efficient algorithms to test for isomorphisms between its input trees.

Since input trees are special instances of (possibly infinite) graphs, we can apply the same notion of isomorphism to compare trees with the same shape:

Definition 5.4 Input tree isomorphism. Two input trees T_i and T_j are *isomorphic* when their graph representations are isomorphic:

$$T_i \simeq T_j . \quad (5.7)$$

That is, there exists an isomorphism $\alpha : T_i \rightarrow T_j$ that maps one input tree to the other one. We call α an *input tree isomorphism*.

Looking back at Fig. 5.6 and Fig. 5.8, we see that $T_2 \simeq T_3$, see also the example of Fig. 5.9. As observed above, if $T_i \simeq T_j$ then i and j are indistinguishable, in the sense that if they start in the same state, then at all times they will always have the same state. This means that nodes with isomorphic input trees are synchronized, provided this pattern is applied to all nodes. This fact can be seen immediately, looking at how T_i was defined:

- Suppose that $T_i \simeq T_j$ and that they are both a one-node tree. Assume these nodes have the same internal dynamic (which is always necessary for synchronization). Then $I_i = \{i\}$, and $I_j = \{j\}$, that is, neither i nor j have any incoming edge: by the anonymity postulate, they start in the same state, so they will remain in the same state forever,

because they cannot receive any external stimulus. They are synchronous (although, of course, this is a trivial synchronization).

- Now, suppose that $T_i \simeq T_j$ and that the roots of these trees have $k > 0$ children. The number of children must be the same, because the trees are isomorphic. Then $I_i = \{i, e_1, \dots, e_k\}$ and $I_j = \{j, e'_1, \dots, e'_k\}$; moreover, the input trees $T_{s(e_1)}, \dots, T_{s(e_k)}$ are isomorphic to the input trees $T_{s(e'_1)}, \dots, T_{s(e'_k)}$, suitably permuted. For the sake of simplicity, let us assume that $T_{s(e_1)} \simeq T_{s(e'_1)}, \dots, T_{s(e_1)} \simeq T_{s(e'_k)}$. By induction, $s(e_l)$ and $s(e'_l)$ will be forever in the same state at the same time, and so will i and j (because i receives stimuli through e_1, e_2, \dots , whereas j receives stimuli through e'_1, e'_2, \dots).

Figure 5.12 shows all input trees for the network of Fig. 5.4, grouping together those that are isomorphic to each other.

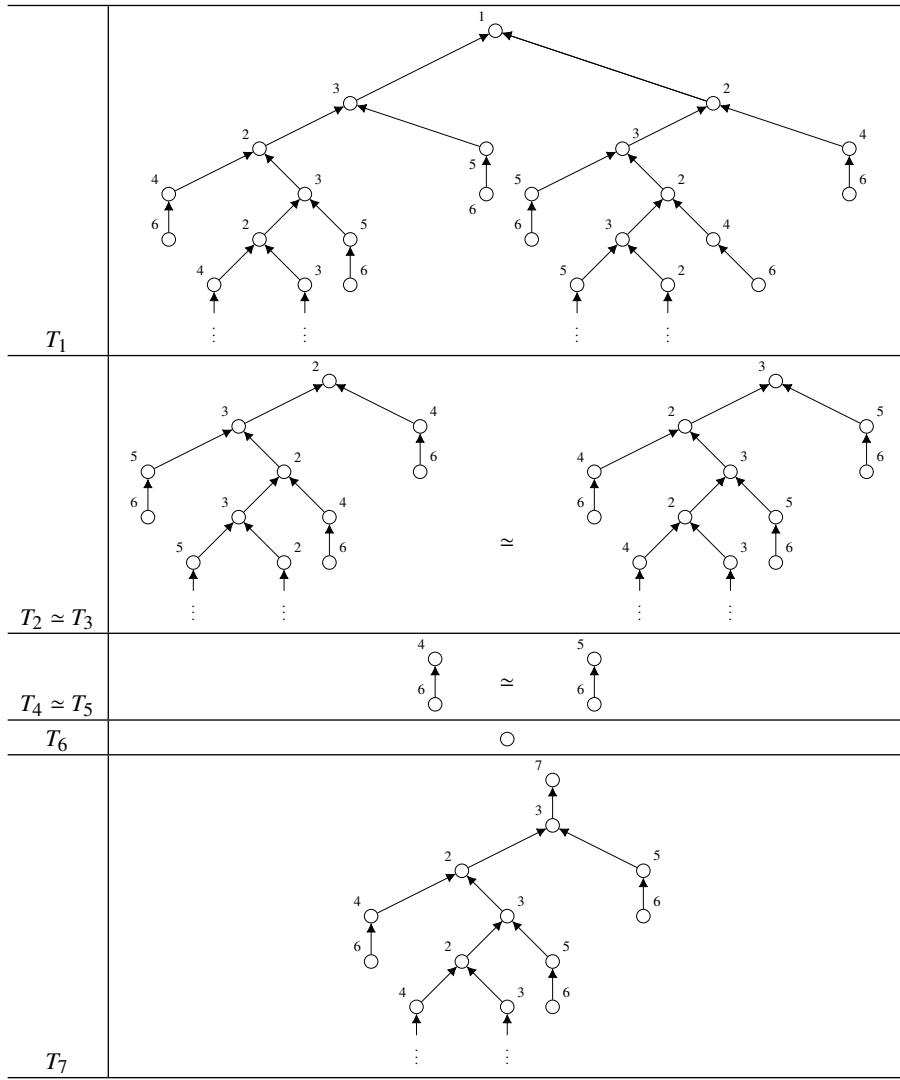
5.7 The graph isomorphism problem: testing input tree isomorphism

The graph isomorphism (GI) problem is the problem of determining whether two finite graphs are isomorphic, that is, whether the graphs are identical despite different drawing and/or labeling (Definition 5.3). Algorithms for the solution of this problem have many practical applications, ranging from chemistry (identification of chemical compounds), computer vision and pattern recognition, to electronic circuit design. The exact complexity of this problem remains one of the most important unsolved problems in theoretical computer science (Grohe and Schweitzer, 2020).

While the exact complexity standing of GI is unknown, it has not been proved to be NP-complete (and it is unlikely to be NP-complete, according to most researchers). There has been recent breakthroughs by Babai (Babai, 2016; Cho, Nov 10; Grohe and Schweitzer, 2020) who proved that GI is almost efficiently solvable, i.e., it is solvable in quasipolynomial time, meaning that Babai's algorithm runs in time as $N^{P(\log N)}$, for some polynomial $p(x)$. Its generalization, the Subgraph Isomorphism Problem, consists of determining whether a graph G has a subgraph that is isomorphic to H , has been proved NP-complete (Cook, 1971).

A breakthrough in practical algorithms for isomorphism testing in large graphs was McKay's Practical Graph Isomorphism (McKay, 1981; McKay and Piperno, 2014), which is quite efficient in practice. It is the basis for McKay's *Nauty* algorithm to find graph automorphisms (McKay, 1990). Indeed, the graph isomorphism problem is computationally equivalent to computing the automorphism group of a graph (Luks, 1993).

Despite GI's complexity being unknown, in many realistic applications or for sufficiently small instances, the problem is easy to solve. For instance, Luks (1982) shows that isomorphism can be decided in polynomial time for graphs with bounded in-degree, and this condition is easily satisfied in all biological networks. Furthermore, here we are concerned only with isomorphisms between input trees, and for trees, GI has efficient, polynomial-time solutions (Kelly, 1957).

**Fig. 5.12**

Examples of input tree isomorphisms. The input trees T_i of all the nodes i of the network of Fig. 5.4 grouping those that are isomorphic.

For our needs, we therefore limit ourselves to isomorphism of trees, which is a polynomial-time solvable problem. Furthermore, in practice, testing for symmetry fibrations does not require testing for input tree isomorphisms. Input trees are a useful theoretical concept, but in most practical applications, we use algorithms to find balanced colorings to obtain the fibers of the symmetry fibration without using the actual input trees. However, input trees are useful conceptually to classify building blocks, see Section 14.5.

As an example, consider again the FFF in Fig. 5.9. Figure 5.9b shows the input trees, and those of nodes 1 and 2 are isomorphic. In this case the isomorphism $\alpha : T_1 \rightarrow T_2$ is a map of nodes as indicated in Fig. 5.9b by the dotted red lines. At the root: $\alpha(1) = 2$; in

the first layer: $\alpha(1) = 1$, $\alpha(3) = 3$, and so on. The edges are also mapped correspondingly: $e_{T_1} = \{1, 1\}$ in the first layer is mapped to the edge $e_{T_1} = \{\alpha(1), \alpha(1)\} = \{1, 2\}$, and so on. The map α is a bijection, because the unique edge connects the image of its source to the image of its target. Therefore, the input trees of Y_1 and Y_2 are isomorphic (i.e., isomorphic as graphs).

These input trees are represented by a graph of infinite size. Do we need to check every layer in the input trees? Obviously, it is not practical to check every edge in an infinite set of edges. However, an important result of Angluin (1980) and Norris (1995) proves that if the first $N - 1$ levels of the input trees are isomorphic (N is the number of nodes in the graph G), then the (infinite) input trees are isomorphic. For example, in the FFF circuit, $N = 3$, so the first two levels of the input tree are enough to show that an isomorphism exists.

Norris's theorem is a fundamental result for fibrations, since it lets us define isomorphism for any input tree, infinite or not, in a finite number of steps. It also has important consequences for interpreting the dynamics, since it limits the number of steps that we need to go back in time to determine the state of the root node in the input tree. If we interpret the input tree as in Fig. 5.4, representing how the information flow of the signals that the root node receives unfold in time, then Norris's theorem implies that the state of the root nodes that are connected to a cycle in the graph is affected for the first time by a bounded number of steps along the cycle. This number of steps is the number of nodes in the circuit. However, a cycle implies a positive feedback on the nodes involved: after the first time a certain node x can impact on the behavior of node y (which depends only on the length of the shortest path from x to y), x will have further influence on y , if x and y are involved together in one (or more) cycles.

Norris's theorem also determines how we define fiber building blocks when the input trees are infinite and determined by cycles in the graph (section 14.5).

5.8 Minimal equitable partition and balanced coloring partition

To take one further step in our analysis, we consider graph partitions, Definition 3.12. We recall: a partition is a way to group nodes into non-overlapping clusters. We visualize a partition by coloring the nodes in each cluster with the same color, using different colors for different clusters.

Of course, there are many possible ways to partition the nodes of a graph. Here, we are exclusively interested in *equitable partitions*, also called *balanced coloring partitions*. Within these partitions we mainly focus on *minimal* equitable and balanced coloring partitions. These correspond to maximal synchronization—fewest clusters.

Definition 5.5 Equitable Partition. Let $\mathcal{S} = S_1, \dots, S_K$ be a partition of the nodes of a network $G = (V, E)$, where K is the total number of parts or clusters in the partition (Definition 3.12). The partition is *equitable* if and only if each node in cluster S_μ has the same number $k_{\nu\mu}$ of incoming edges from nodes in cluster S_ν , for $1 \leq \mu, \nu \leq K$.

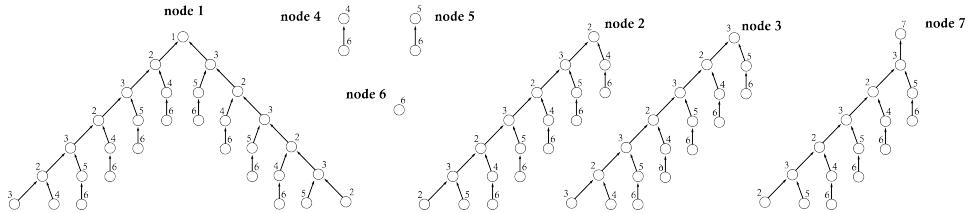


Fig. 5.13

Isomorphic input trees for the minimal equitable partition. Input trees of the graph shown in Fig. 5.4. We find: $T_2 \simeq T_3$ and $T_4 \simeq T_5$. Grouping these two clusters lead to the minimal equitable partition of Fig. 5.14. If we do not group all isomorphic input trees but just some of them, then the resulting partition is still equitable but not minimal, see Fig. 5.15. Grouping nodes that are not isomorphic results in a partition that is not equitable, see Fig. 5.16.

Equitable partitions and (robust) cluster synchronization are logically equivalent concepts (Golubitsky and Stewart, 2023, Theorem 10.18). We think of an equitable partition as a way to represent a cluster of synchronous states of the system. Nodes in the same partition have the same (local) state, i.e., they synchronize in the cluster. Nodes that are in different clusters may have different (local) states or different synchronous states. Equitable partitions are defined by input tree isomorphisms. We use colors to visualize these partitions: colors represent local states, and the clusters of the partition are the groups of nodes with the same color.

In general there are many equitable partitions, and enumerating them all is computationally challenging. However, several algorithms to do this exist, for example (Kamei and Cock, 2013a; Belykh and Hasler, 2011); see Chapter 13. But one partition is of particular interest here because it collects all the symmetries that the graph has. This is the *minimal equitable partition* or *coarsest balanced coloring*, and it partitions the graph into the smallest number of clusters.

Definition 5.6 Minimal equitable partition. The *minimal* equitable partition is the (unique) equitable partition with the minimal number of clusters (colors) or the *coarsest* equitable partition that can be found. It can be proved to be the equitable partition where two nodes are in the same cluster if and only if they have isomorphic input trees.

Figure 5.13 shows the input tree isomorphisms of the graph of Fig. 5.4. Input trees of node 2 and 3 are isomorphic, $T_2 \simeq T_3$, and nodes 4 and 5 too, $T_4 \simeq T_5$. There are no more isomorphisms as shown by the rest of the input trees. If we group *all* isomorphic nodes into clusters we obtain the minimal equitable partition with 5 colors shown in Fig. 5.14. This equitable partition is minimal because all nodes with isomorphic trees are in the same cluster.

If we break one of these clusters and assign two different colors, for instance, to nodes 2 and 3, but do not break synchrony of nodes 4 and 5, the partition is still equitable but not minimal, since it has 6 colors rather than 5, see Fig. 5.15. Note, however, that in general if

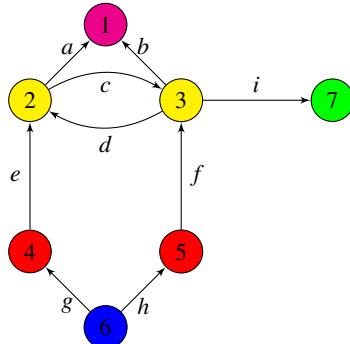


Fig. 5.14

Minimal equitable partition. The graph of Fig. 5.4 with nodes colored according to their input trees. Two nodes have the same color if and only if their input trees are isomorphic, which specifies the minimal equitable partition.

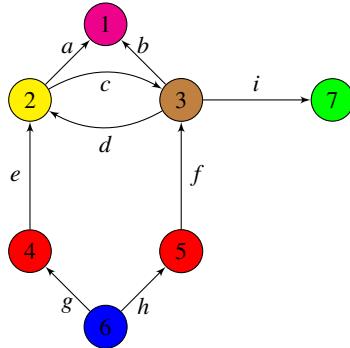


Fig. 5.15

Equitable partition that is not minimal. When we do not group all the nodes with isomorphic input trees with the same color, for instance, node 2 and 3 are isomorphic but we assign them different colors, we end up with an equitable partition that is not the minimal one, since we are using two colors to color what previously was one cluster.

we take one cluster from an equitable partition and break it arbitrarily, the resulting partition may not be equitable. For instance, if we break the synchrony between 4 and 5 in Fig. 5.14, then also the synchrony between 2 and 3 ceases to exist. On the other hand, if we break *both* clusters we obtain again an equitable partition (a trivial one in which each cluster contains just one node).

We have established that all equitable partitions (not just the minimal one) group nodes with isomorphic input trees. An equitable partition has another special property, which not all partitions share. If you look at the *input sets* (not the input trees) of all nodes, and color them according to the equitable partition, one important property holds true: if two nodes have the same color, then their input sets are color-isomorphic (that is, isomorphic and with an isomorphism that respects the colors). This means that nodes with the same color receive the same colors from their respective input nodes. For instance, in Fig. 5.14, nodes

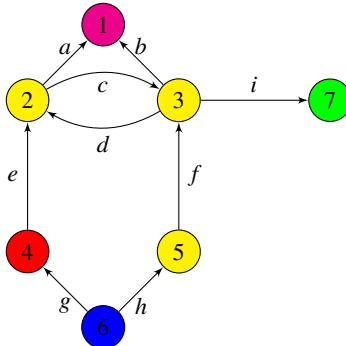


Fig. 5.16

Partition that is not equitable. If we color the graph of Fig. 5.4 by grouping nodes with the same color that do not have isomorphic input trees, for instance yellow for node 5 which is not isomorphic with 2 and 3, the resulting partition is not equitable. The coloring represents a non-equitable partition (nodes with the same color have different color-isomorphic input sets).

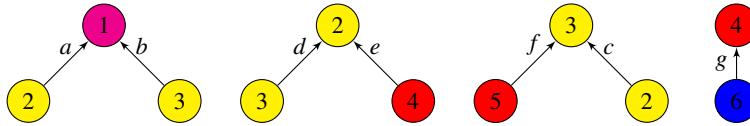


Fig. 5.17

Color-isomorphic input sets in the equitable partition. The input sets of Fig. 5.4 (shown in Fig. 5.5) but, here, colored according to the coloring of Fig. 5.14: the coloring represents an equitable partition (nodes with the same color have color-isomorphic input sets).

2 receives one red color and one yellow color, and node 3 as well, although it receives a red from a different node. Then, nodes 2 and 3 are colored the same. Color-isomorphism between input sets is the main concept that connects the fibration formalism with the groupoid formalism of Golubitsky and Stewart (Golubitsky and Stewart, 2006), which is treated in detail in Chapter 7.

This fact is clear in Fig. 5.14. The input sets I_1 , I_2 and I_3 are all isomorphic, because nodes 1, 2, 3 have all exactly two incoming edges, see Fig. 5.5. However, after coloring we see that only I_2 and I_3 are color isomorphic, see Fig. 5.17.

This observation leads to an alternative way to define an equitable partition without using the full input tree, but just using its first layer, i.e. the input set. This may seem an improvement, since we do not need to consider (possible) infinite trees, but the drawback is that we need to know the colors first. For this definition to work, we must first find the equitable partition of the nodes, and then test color-isomorphism of the input sets. This problem disappears if we use the input tree to define the equitable partition, Definition 5.6. However, we obtain only the minimal equitable partition in this manner.

Balanced coloring partition

Let us formalize these statements. An equitable partition (Definition 5.5) is an equivalent concept to a balanced coloring partition, defined below. These terms are used in different areas, and have a slightly different emphasis. The usual definition in the formalism of (Golubitsky and Stewart, 2023) is:

Definition 5.7 Balanced coloring partition. A *coloring* κ of a graph $G(V, E)$ assigns a color $\kappa(c)$ to each node c , where $\kappa(c)$ belongs to a set \mathcal{K} , called the set of *colors*. The coloring κ is *balanced* if $\kappa(c) = \kappa(d)$ implies that $I(c)$ and $I(d)$ are color-isomorphic.

Equivalently, a balanced coloring is a coloring such that each node with color μ in cluster S_μ is connected (by arrows of the same type) to the same number of nodes $k_{\mu\nu}$ with color ν , for $1 \leq \mu, \nu \leq K$ where K is the number of colors.

This definition is essentially the same as that of an equitable partition, Definition 5.5, but in different terminology. There is also a natural minimal balanced coloring partition, which is the same as the minimal equitable partition (Golubitsky and Stewart, 2023, Theorem 13.16):

Definition 5.8 Minimal balanced coloring partition. This is a balanced coloring of a graph with the minimal number of colors. It is the same as the minimal equitable partition.

In terms of synchronization, nodes inside the same subset S_i of the balanced coloring partition can synchronize, since they receive the same color inputs from the same synchronized nodes. Thus, color balance is another way to say that the partition is equitable.

An equitable partition represents a global state that has a robust synchrony pattern. In fact, equitability means that if two nodes (say i and j) are currently in the same state, they read inputs from sources in the same state, so they will never diverge from each other. Conversely, in non-equitable partitions this behavior does not take place: in the global state of Fig. 5.16, nodes 2 and 3 are both in the state yellow, but one receives inputs from two yellow nodes, and the other receives inputs from one yellow and one red, so 2 and 3 may well change state. For a rigorous proof, see (Golubitsky and Stewart, 2023, Proposition 10.20).

And what about our old friend, the orbital partition from Definition 4.6? As discussed in section 4.2, the orbital partition also produces a partition of synchronized nodes, and it is, by definition, equitable and color-balanced. The origin of synchronization in the orbital partition is the existence of automorphisms in the graphs. Automorphisms determine certain special fibrations, and are the best-known way to create balanced colorings. But, as we have already seen, not all equitable partitions are orbits of subgroups of automorphisms of the network, because there can be additional fibrations. In particular, the minimal orbital partition need not be the minimal equitable partition.

In fact, throughout this book, we deal almost exclusively with equitable partitions that are not orbital, that is, equitable partitions that do not arise from automorphisms of the network, but from a symmetry fibration.

The main difference is that balanced and equitable partitions are about fibrations and

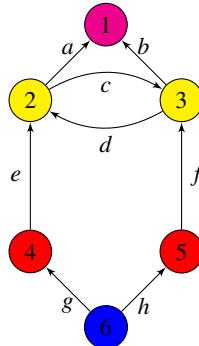


Fig. 5.18

Example of orbital partition that is also the minimal equitable partition. A graph with a left-right automorphism that induces the shown orbital partition. There are no extra symmetries in this graph, so the orbital partition is the most symmetric one. The orbital partition is the minimal equitable partition and fibrations do not add much information to this analysis.

groupoid symmetry, which are local concepts. Orbits are about global group symmetry. In general, there is very little reason for a local symmetry to extend to a global one, which opens the door to a much richer world of symmetries than those represented by orbits and automorphisms.

5.8.1 Examples of balanced coloring partitions

We can continue our journey by providing some examples. We are mainly interested in the minimal partition, which captures most of the symmetries.

Example 3 Figure 5.18 shows a case where the orbital partition is indeed the minimal equitable partition. That is, all the symmetries of the graph are captured by its automorphisms, and fibration symmetries do not add much to explain this structure. We studied this graph in Fig. 5.3a and concluded that it has left-right symmetry. This induces the orbital partition shown in Fig. 5.18. This is an equitable partition and indeed the minimal one, as the input trees show.

Example 4 However, we have shown that the addition of a single outgoing edge from node 3 to node 7 as in Fig. 5.14 destroys this global automorphism, and the only automorphism of the modified graph is the identity, so the graph is rigid. Consequently, the orbital partition is now reduced to the most trivial one (see Fig. 5.19): the identity partition where each node has a different color. This partition is also equitable: in this case the definition of equitable (if two nodes have the same state then . . .) is trivially true because no two nodes have the same state. This trivial partition requires seven colors (one color per node), and it is the least coarse (the *finest*), the one with maximal number of colors, and farthest from the minimal equitable partition of five colors shown in Fig. 5.14.

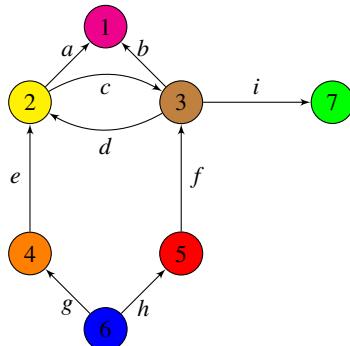


Fig. 5.19

Example of trivial orbital partition in a graph with nontrivial minimal equitable partition. Adding the single edge to node 7 from Fig. 5.18 reduces the orbital partition to the trivial one: one color per node. This trivial orbital partition is far from the minimal equitable partition for this graph shown in Fig. 5.14. This represents an example where fibration symmetries are necessary to capture the symmetries of the graph. It is ubiquitous in biological networks.

Example 5 An intermediate example of an undirected graph with coexistence of an orbital partition that is not a minimal partition is shown Fig. 5.20. The graph has only a global up-down symmetry, creating the six orbits in pairs of nodes as shown in Fig. 5.20a. However, these are not the only symmetries of this graph, and the orbits do not capture all the clusters of synchronous states. While the graph has no global left-right symmetry, locally, there are extra left-right symmetries creating extra synchronous states by the minimal equitable partition shown in Fig. 5.20b.

All green nodes are connected to two red nodes and one blue node. All red nodes are connected to two green nodes and one red node, and all blue nodes are connected to two green nodes, creating a more symmetric minimal balanced coloring than the orbital partition with just three colors. The green nodes (and blue nodes) are symmetric in terms of their local topology despite the lack of global left-right symmetry. The coloring represents a left-right symmetry through the conservation of color in a local neighborhood of each node, i.e., the left-right mirror transformation conserves the balanced coloring. The symmetry in local topology captured by the minimal balanced coloring is less restrictive (and more perspicuous) than the global one.

Example 6 Frucht Graph.

A notable example, and a more extreme one in some sense, is the Frucht graph described by Robert Frucht in 1939 (Wikipedia, 2022) and shown in Fig. 5.21. Frucht is better known for his influential theorem stating that any group can be realized as the group of symmetries of a graph. Even more interesting is his subsequent theorem stating that all groups can be represented as the automorphism group of a 3-regular graph (or cubic graph, a graph in which all nodes have degree three) (Frucht, 1949).

The Frucht graph is an undirected graph with all degrees equal to 3 (undirected edges

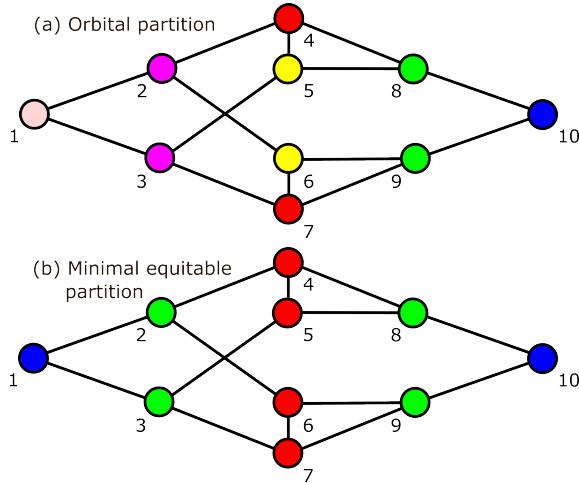


Fig. 5.20

Example of a graph whose orbital partition is not the minimal equitable partition.

(a) Orbital partition of the graph with six colors arising from the up-down automorphisms of the graph. (b) Minimal equitable partition with three colors, showing extra symmetries not captured by the orbits. In fact, the graph has no left-right symmetry, yet the green and blue clusters are still fibration symmetric in their local topology.

are interpreted as bidirectional, i.e., each edge can be seen as a pair of arrows in opposite directions). It is famous because, despite the apparent regularity of its structure, it has no nontrivial automorphisms: its only automorphism is the identity, hence the only orbit coloring is the one in Fig. 5.21e, where all nodes have a different color.

Although its automorphism group is trivial, the Frucht graph has a minimal equitable partition with just one color that represents the most symmetric state (Fig. 5.21a). This is possible because the graph is *regular*: all nodes have the same degree and receive the same unique color from adjacent nodes. The Frucht graph can therefore display complete synchronization under any admissible dynamics.

To study whether the Frucht graph has any further nontrivial balanced colorings, we can use the combinatorial characterization of a balanced coloring. This requires nodes of the same color to have correspondingly colored inputs, for suitable permutations of the input arrows. The analysis leads to precisely five different balanced colorings, shown in Fig. 5.21. Only coloring (e) is given by the symmetry group, which is trivial. The other four balanced colorings do not arise from automorphisms. Coloring (a) is obvious from the regularity of the graph, and it constitutes the minimal equitable partition. The other three are far from obvious, but can be found by various methods that we do not enter into here. The algorithms discussed in Chapter 13 show that these colorings are the only balanced colorings of the Frucht graph.

Any n -node 3-regular graph has $6n^2$ fibration symmetries. Here $n = 12$, giving 864

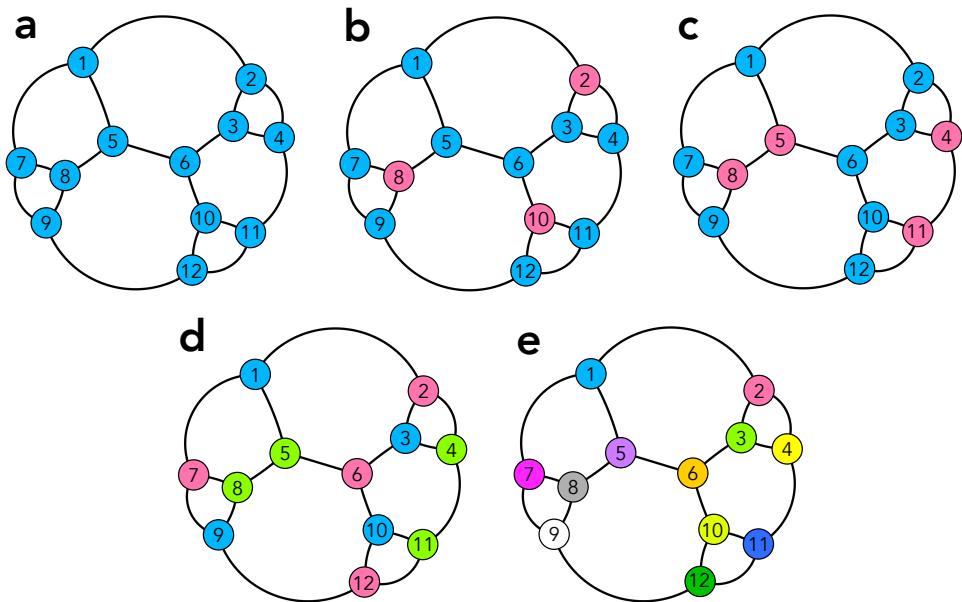


Fig. 5.21

The amazing hidden symmetries of the Frucht graph. Complete list of balanced colorings of the Frucht graph, assuming all edges are bidirectional. (a) Full symmetry: All nodes have identical colors, leading to complete synchrony. (b) Two colors: each red node has three blue inputs; each blue node has one red and two blue inputs. (c) Two colors: each red node has one red and two blue inputs; each blue node has one red and two blue inputs. (d) Three colors: each blue node has two red and one green input, each red node has two blue and one green input, and each green node has one red, one blue, and one green input. (e) All nodes are different: trivial coloring.

fibration symmetries, compared to 1 group symmetry. Nevertheless, these fibration symmetries lead to only 5 equitable partitions. This shows that the relation between fibration symmetries and equitable partitions is subtle.

Example 7 Regular graphs. More generally, any k -regular graph (i.e., any undirected graph whose nodes have exactly k neighbors) with N nodes can be colored with one single color (we will see later that it can be fibered over a k -bouquet: a one-node graph with k loops). This coloring corresponds to the highest symmetry of the graph: all nodes are the same. This contrasts with the conclusion that would arise by considering only automorphism symmetries.

From the point of view of automorphisms, it is known that for $k \geq 3$ almost all k -regular graphs are rigid (have no automorphisms except for the identity). ‘Almost all’ here means that the fraction of all k -regular graphs with N nodes that are rigid tends to 1 as $N \rightarrow \infty$. In other words, the Frucht graph is just an example of a more general and almost unavoidable phenomenon. Although these graphs have no nontrivial automorphisms, and in

principle every node can be distinguished topologically from every other node, these graphs have the most symmetric balanced coloring possible: all nodes are the same, and in fact, they can synchronize in unison under any admissible dynamics. Moreover, any synchrony pattern determined by a balanced coloring can be realized as a *stable* equilibrium of *some* admissible ODE (Golubitsky and Stewart, 2023).

To move forward we need to abandon the notion of an automorphism as the only useful symmetry notion, and embrace the more general form of symmetry of equitable partitions and fibrations. Only then we will be able to describe relevant symmetries of biological networks.

Remark The balanced coloring problem should not be confused with the ‘vertex coloring problem’, sometimes referred to as just the ‘coloring problem’ or ‘graph coloring’. The vertex coloring problem attempts to find a coloring of the graph such that no two adjacent nodes have the same color. The balance condition is different. These two problems are unrelated.

5.9 Robust versus fragile synchrony

In Section 2.5, we stated a key principle for this book:

$$\begin{aligned} \text{Synchronized nodes must have the same internal dynamics} \\ \text{and must receive synchronized signals.} \end{aligned} \tag{5.8}$$

This principle, which is highly intuitive, led to the conclusion that synchrony clusters determine a fibration/balanced coloring. However, in Section 5.3.1 we remarked that this deduction requires an extra technical condition to be mathematically valid. In this section we expand on this remark. This discussion is a mathematical fine point and can be skipped.

We begin with an example to illustrate the mathematical phenomenon involved.

Example 8 Consider the 5-node bidirectional graph in Fig. 5.22a. Here nodes 1, 2, 3, and 4 are input isomorphic, with in-degree (or valence) 3. Node 5 has in-degree 4, so it is not input isomorphic to the other four nodes. The coloring in Fig. 5.22b is balanced. However, the colorings in Fig. 5.22c, d are not balanced, because nodes with the same color must be input isomorphic in any balanced coloring.

Now consider a discrete reaction-diffusion equation(5.9), where f is any reaction function and a is a diffusion coefficient.

$$\begin{aligned} \dot{x}_1 &= f(x_1) + a(x_1 - x_2) + a(x_1 - x_4) + a(x_1 - x_5) \\ \dot{x}_2 &= f(x_2) + a(x_2 - x_1) + a(x_2 - x_3) + a(x_2 - x_5) \\ \dot{x}_3 &= f(x_3) + a(x_3 - x_2) + a(x_3 - x_4) + a(x_3 - x_5) \\ \dot{x}_4 &= f(x_4) + a(x_4 - x_1) + a(x_4 - x_3) + a(x_4 - x_5) \\ \dot{x}_5 &= f(x_5) + a(x_5 - x_1) + a(x_5 - x_2) + a(x_5 - x_3) + a(x_5 - x_4). \end{aligned} \tag{5.9}$$

This is an admissible ODE for the graph. By general theory, coloring (b) gives a consistent

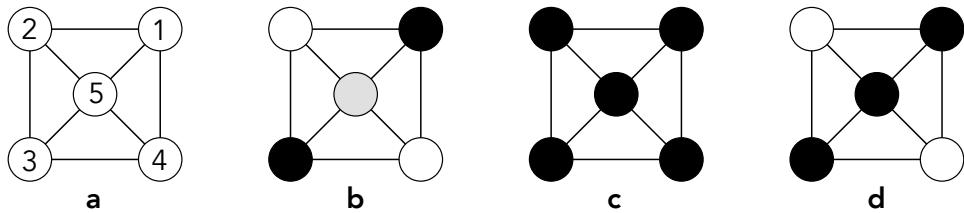


Fig. 5.22

Colorings of a 5-node undirected graph. (a) The original graph. (b) A balanced coloring. (c) The coloring for complete synchrony is not balanced. (d) This coloring is also not balanced. However, all three colorings correspond to cluster synchrony in the model ODE (5.9).

equation for cluster synchrony, as can be checked directly by setting $x_1 = x_3 = x$, $x_2 = x_4 = y$, $x_5 = z$. Although coloring (c) is not balanced, substituting $x_1 = x_2 = x_3 = x_4 = x_5 = x$ leads to five copies of the same equation $\dot{x} = f(x)$. There is no inconsistency, and any solution of that equation yields a solution of (5.9) in which all five nodes are synchronous. Similarly, although coloring (d) is not balanced, substituting $x_1 = x_3 = x_5 = x$ and $x_2 = x_4 = y$ leads to

$$\begin{aligned}\dot{x} &= f(x) + 2a(x - y) \\ \dot{y} &= f(y) + 3a(y - x)\end{aligned}$$

and again, there is no inconsistency

At first sight, the cluster synchrony patterns (c) and (d) contradict our claim that principle (5.8) implies that every synchrony pattern corresponds to a balanced coloring. It does not contradict the principle itself: nodes x and y do indeed receive synchronized inputs, *for this ODE*. However, as Example 8 shows, the inference of a balanced coloring can have exceptions. In this case, the exceptional patterns arise from the special nature of the coupling terms, which vanish when the nodes concerned are synchronous. These interactions are, therefore, ‘invisible’ to the dynamics. In fact, these types of couplings are the diffusive couplings leading to the Laplacian dynamics described in Section 3.6.1. They are a popular way to describe complete synchronization patterns, but we see here that in terms of cluster synchronization, they are fragile patterns of the admissible ODEs.

To avoid such exceptions (which are ‘rare’ in a suitable context, here admissible ODEs) we can modify (5.9) by imposing the extra condition of *robustness* (Stewart et al., 2003). This was mentioned in Section 5.3.1 and is defined as follows:

Definition 5.9 A cluster synchrony pattern for a given graph and admissible ODE, is *robust* if that pattern, when it is imposed upon *any* admissible ODE, leads to a consistent ODE for the cluster dynamics.

A cluster synchrony pattern that is not robust is *fragile*.

It is easy to see that adding an extra small term such as εx_5 to the component of (5.9) for \dot{x}_5 , which preserves admissibility, leads to two conflicting equations for \dot{x} in patterns

(c) and (d). Thus these patterns are fragile. Of course, if diffusive coupling is a sensible modeling assumption, such fragility would not be relevant for biological purposes.

The term ‘robust’ should not be confused with less formal uses of the word. In this section we use it in this technical sense, which goes back to Stewart et al. (2003), in order to clarify when the relation between cluster synchrony and balanced colorings holds. Robustness in this technical sense carries no other connotations that might be associated with that word. In particular, it does not imply the stability of the cluster state concerned, only its existence. Stability depends on details of the model ODE, and it is discussed in Chapter 8. Section 8.12 further discusses this notion of robustness and compares it with others in the literature. The implication of the robustness Definition 5.9 for the existence of synchrony and the representation of the system is further discussed in Section 9.3.

The central point of this example, and others like it, is that input isomorphisms are properties of the *equation*, not just of some synchronous *solution*.

Remark In fact, a less restrictive condition also ensures that cluster synchrony corresponds precisely to fibrations/balanced colorings. The Rigid Equilibrium Theorem (Golubitsky and Stewart, 2023, Theorem 14.9) shows that if clusters of an equilibrium state persist after any *sufficiently small* admissible perturbation, then the corresponding coloring must be balanced. This formulation is highly relevant to biology. It tells us that if any *small* change to the model equations that is consistent with the graph structure does not change the clusters—which is a reasonable requirement, since most biological models lack the precision of physical ones—then the clusters must arise from a fibration.

A similar result is conjectured for all networks when the dynamics is time-periodic. This has been proved for a wide variety of networks, and for all networks under stronger hypotheses: see (Golubitsky and Stewart, 2023, Chapter 15). Something similar should usually apply to chaotic states as well, but the issues involved are outside the scope of this book.

The key point is that fibrations/balanced colorings are fundamental to the existence of meaningful cluster states, *unless* the area of application uses specific modeling assumptions that allow unbalanced colorings to occur naturally.

5.10 Comments on equivalent definitions in different disciplines

Now that we have established that ‘equitable partition = balanced coloring’, it is worth asking why we are providing two definitions for the same concept. The answer is that the literature has grown from several quasi-independent sources, in different disciplines. Anyone consulting these sources needs to be aware of differences in viewpoint and terminology.

The term ‘equitable partition’ is typically used in the computer science literature where the graph fibration formalism was originally developed by Boldi and Vigna (2002a). On the other hand, the name ‘balanced coloring’ is used in the dynamical systems and mathematics

communities where the groupoid formalism was originally developed in (Stewart et al., 2003; Golubitsky et al., 2005a).

Both concepts are mathematically equivalent. The overlap of these concepts across various subject areas from computer science, mathematics, dynamical systems, chaos and physics makes this topic multidisciplinary. This introduces a problem of communication between these disparate disciplines. Unfortunately, the fibration literature from computer science and the balanced coloring literature from dynamical systems hardly overlap, even though they speak about the same concepts. One of the goals of this book is to bring these communities together by providing common definitions and concepts. Figure 1.4 is an attempt to do just that. Sometimes, it is just a matter of language, as when one community calls an equitable partition what the other calls a balanced coloring. Sometimes, however, concepts need to be related in less obvious ways.

To make things more complicated, we are targeting another community with its own jargon: physicists. In physics, symmetries are synonymous with symmetry groups. In particular, particle physics and general relativity are built on fiber bundles (discussed in Chapter 10). Fibrations are generalizations of fiber bundles, and following physics jargon, we refer to the synchronous clusters in equitable/colored partitions as the *fibers* of the graph (see section 6.4) in analogy with fiber bundles in physics.

We may finally wonder: where does biology enter into this picture? It has not entered anywhere so far, because fibrations have not previously been discussed in biology, so there is no biological jargon related to fibrations to worry about. However, ‘fiber’ has other meanings in biology; this should not cause confusion. This book is an attempt to combine all previous results to build a new language for the application of fibrations to biological networks.

We now focus on the central concept of this book: that of a *symmetry fibration* or *fibration symmetry*. We define (graph) homomorphisms, which ‘collapse’ nodes and edges in a consistent manner, and (graph) fibrations, which are homomorphisms that also preserve input sets. We relate fibrations to equitable partitions, which are equivalent to balanced colorings of the nodes. We discuss the key property of a fibration, the lifting property. We emphasize the role of minimal equitable partitions (minimal balanced colorings), which employ the smallest number of colors and group the nodes into the fewest clusters. Finally, we consider non-minimal colorings.

So far, we have established that equitable partitions describe robust symmetries between the nodes of a network. A more perspicuous way of describing equitable partitions is by means of a graph fibration, which is a special form of graph homomorphism. Next, we define a graph homomorphism, and then we show that the minimal equitable partition emerges from a symmetry fibration of the graph.

6.1 Graph homomorphisms

Since equitable partitions, balanced colorings and cluster synchronization are all about colored source nodes of incoming edges captured by the input trees, it is natural that the maps that define these partitions should preserve certain incidence relations among the nodes. Maps that preserve incidence relations are called homomorphisms, and a fibration is a particular kind of graph homomorphism that further preserves a particular kind of incidence, the lifting property. To understand fibrations we must first understand homomorphisms (Boldi and Vigna, 2002a). They are similar to graph isomorphisms, Definition 5.3, but the map concerned need not be a bijection.

Recall that an undirected graph can be viewed as a directed graph in which all arrows come in pairs, pointing in opposite directions. This lets us focus on directed graphs without losing generality. It is also a natural convention for admissible dynamics.

A graph homomorphism is a map from nodes to nodes and edges to edges that preserves source and target of every arc (Harary, 1969; Cameron, 2006; Hahn and Tardif, 1997; Hell and Nesetril, 2004); in other words, if it sends some arc a of G to some other arc a' of H , then it must also send the source (target) of a in G to the source (target) of a' in H . Formally:

Definition 6.1 Graph homomorphism. Given two graphs $G = (N_G, E_G)$ and $H =$

(N_H, E_H) , a *graph homomorphism* $\alpha : G \rightarrow H$ is a map from G to H such that if i and j are adjacent in G via an edge, then $\alpha(i)$ and $\alpha(j)$ are adjacent in H by an edge in the same direction.

In other words, a graph homomorphism α maps nodes of G to nodes of H and edges of G to edges of H in such a way that the image of the edge connects the image of the source and the image of the target of this edge. This requirement is very minimal, and the existence of a homomorphism from G to H does in general not imply that G and H are in any way similar.

In particular, a homomorphism need not be an isomorphism; it is a more general concept that encompasses also automorphisms and fibrations, as well as other maps between graphs that are not fibrations. A homomorphism is a weak form of incidence preservation: If there is more than one edge between i and j , then it requires at least one between $\alpha(i)$ and $\alpha(j)$. A fibration is a stricter form of homomorphism since not only preserves incidence but it also preserves the input trees (via the lifting property) and therefore preserves the dynamics between G and H , which is not necessarily preserved by the homomorphism.

Multigraphs (with parallel edges, Definition 3.9) require a generalization:

Definition 6.2 Multigraph homomorphism. A *multigraph homomorphism* is a pair of functions $\alpha_N : N_G \rightarrow N_H$ and $\alpha_E : E_G \rightarrow E_H$ such that:

$$s_H(\alpha_E(a)) = \alpha_N(s_G(a)) \quad \text{and} \quad s_H(\alpha_E(a)) = \alpha_N(s_G(a)). \quad (6.1)$$

For typed (multi)graphs, it is required that types of nodes and edges are respected: if a node i (or an arc a) has a certain type, then its image $\alpha_N(i)$ ($\alpha_E(a)$, respectively) must have the same type.

In various branches of abstract algebra, the analogs of graph homomorphisms are called just homomorphisms, which are structure-preserving maps between two algebraic structures or sets of the same type (group, ring, module ...). Vector space homomorphisms are called linear maps, for historical reasons. This notion is then further generalized in category theory, where they are called *morphisms*. Homomorphisms are important since they encompass the definitions of isomorphisms, automorphisms and fibrations as special cases. Despite their similar names, homomorphisms are different from *homeomorphisms*, which arise in topology and preserve topological structures, like deforming a coffee mug into a donut (torus).

Graph homomorphisms are maps, and any map can be injective, surjective, bijective (injective and surjective), or neither (see Fig. 6.1). We recall these definitions since they will be important for the types of fibrations that we use to describe biological graphs. For more details see (Stewart and Tall, 2015).

Definition 6.3 Injective map. A map is *injective* if it maps distinct elements of its domain to distinct elements of its codomain (Fig. 6.1a).

Therefore, every element of the codomain is the image of at most one element of its

domain. An injective map is also called a ‘one-to-one map’. An injective homomorphism (i.e., one that maps injectively both nodes and edges) can map a graph to a bigger graph. Even though the map is one-to-one, there could be elements in the codomain that are not images of any elements in the domain. Thus, a one-to-one map need not be a correspondence (as in a bijection), and an injective map need not be invertible.

When dealing with (multigraph) homomorphisms (Definition 6.2), injectivity is required both for the node map and for the edge map. More precisely, a multigraph homomorphism $\alpha : G \rightarrow H$ is injective if and only if both α_N and α_E are injective.

Definition 6.4 Surjective map. A map $\alpha : G \rightarrow H$ is *surjective* if for every element of the codomain of $y \in H$, there is an element in the domain $x \in G$ such that $\alpha(x) = y$ (Fig. 6.1b).

That is, in a surjective map every element of the codomain is the image of at least one element in the domain. For this reason, a surjective map is also called ‘onto’. Such a map may lead to a reduction of the domain through the ‘collapse’ of some elements, and will be the main type of map studied here.

Again, when dealing with (multigraph) homomorphisms (Definition 6.2), surjectivity is required both for the node map and for the edge map. More precisely, a multigraph homomorphism $\alpha : G \rightarrow H$ is surjective if and only if both α_N and α_E are surjective. Surjective fibrations, which collapse nodes or edges, are the most common fibrations discussed in this book, since they describe cluster synchronization and dimension reduction in biological models.

Definition 6.5 Bijective map. A *bijection* is a map that is both injective and surjective (Fig. 6.1c).

A bijection is a map such that every element in the codomain is an image of exactly one element in the domain. Thus, a bijection is both injective and surjective. A bijection is a ‘one-to-one correspondence’ between two sets, and it is invertible.

When dealing with (multigraph) homomorphisms (Definition 6.2), bijectivity is required both for the node map and for the edge map, as usual.

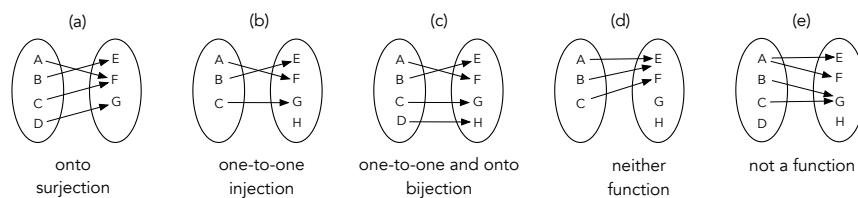


Fig. 6.1

Types of maps. (a) Injective. (b) Surjective. (c) Bijective. (d) Neither injective nor surjective. (e) Not a function.

When the map preserves some kind of algebraic structure, an injection is often called a ‘monomorphism’, a surjection is an ‘epimorphism’, and a bijection is an ‘isomorphism’.

Over the years, different communities have developed different terms for the same concept, and mathematical fashions have changed.

We have seen already two examples of bijective homomorphisms, namely isomorphisms and automorphisms, which are compared to each other in Fig. 5.10. Injective homomorphisms are not very relevant for biological networks since they lead to larger graphs (although an inverse injective fibration will be used in Chapter 19 to reduce the complexity of the network). The relevant types of homomorphism for biological applications are the surjections, and within those the surjective fibrations. Fibrations are a very special class of homomorphism with an additional condition: preserving color-isomorphic input sets by satisfying the lifting property. We discuss this concept next.

6.2 Graph fibrations

A fibration is a homomorphism that preserves more adjacency structure than the simple incidence relation: the exact definition is related to the so-called lifting property, and we postpone it to Section 6.5. The basic idea of fibrations, though, is that they can collapse only nodes in synchronous fibers, i.e., nodes in equitable partitions. Here ‘collapse’ means ‘map to the same thing’, that is, ‘identify’. Equivalently, they collapse nodes with isomorphic input trees or with color-isomorphic input sets. This is why we believe that fibrations are the right type of graph homomorphism to describe biological networks.

6.3 Examples of fibrations

Before giving the general definition of a graph fibration we consider two examples.

Example 9 Metabolator and Smolen Circuits revisited. We return to the metabolator and Smolen networks of Section 5.1, Fig. 5.1, each of which has two nodes and two arrow types—biologically, activation and repression. We saw that both circuits support completely synchronous states, in which nodes 1 and 2 synchronize; see Fig. 6.2 (top).

For the metabolator, this state is the minimal orbital partition for its automorphism group \mathbb{Z}_2 . The Smolen circuit has trivial automorphism group, and its minimal orbital partition is trivial, with parts $\{1\}$ and $\{2\}$. The minimal equitable partition has a single part $\{1, 2\}$. This occurs because the nodes have color-isomorphic input sets: each pink node receives one pink input of each arrow-type.

We now recast this analysis in the language of fibrations. Both networks have fibrations to the *base* illustrated in Fig. 6.2 (bottom). The base, which is the same for both circuits, has a single node, and receives one input of each arrow type from itself.

The colored arrows indicate the corresponding fibrations. Red arrows show how the nodes map (both nodes 1 and 2 map to node X in the base). Blue arrows show how the activator arrows map; green arrows show how the repressor arrows map. The key feature

that makes this map a fibration is that it preserves input sets. Each node, be it 1, 2, or X, has one activator input and one repressor input.

There is a unique fiber in each case, namely the set {1, 2} of all nodes.

Example 10 Next, we consider a less trivial example of a fibration using the FFF circuit depicted in Fig. 6.3 (center). We show in yellow nodes 2 and 3, which belong to the same part of an equitable partition: they have isomorphic input sets, or equivalently isomorphic input trees. We apply to the FFF graph (Fig. 6.3a) three surjective homomorphisms: Fig. 6.3b, a surjective homomorphism collapsing nodes of 1 and 2 (both mapped to node 3'); Fig. 6.3c, a surjective homomorphism collapsing nodes of 1 and 3 (both mapped to node 1'); and Fig. 6.3d, a surjective fibration collapsing nodes 2 and 3 (both mapped to 2'). We also show in Fig. 6.3e a collapsed graph that is obtained with a surjective node map but not a homomorphism.

A homomorphism preserves adjacency: edges are mapped in such a way that the image of the edge connects the image of the source to the image of the target, transferring the incidence relation from source to target node in G to their respective images in H .

In a homomorphism, it can happen that nodes are collapsed (i.e., they are mapped to the same image); because of its very definition, the homomorphism ‘drags along’ the edges of the collapsed nodes to H , enough to have at least one edge satisfying the preservation of adjacency.

In practice this means that when we collapse nodes 1 and 3 into the single node 1' ($\varphi(1) = \varphi(3) = 1'$) in the surjective homomorphism of Fig. 6.3c, node 3 drags its incoming edges into the collapsed node 1': this is achieved by mapping $c \rightarrow c'$ and $b \rightarrow b'$, such that both incoming edges of 3 are transferred to 1'. Thus, through the edge map $\varphi(c) = c'$, we see that the image (c') of the edge (c) connects the image of the source $\varphi(s(c))$ and the image of the target $\varphi(t(c))$ of the source edge c , thus preserving incidence. The same occurs for the edge b , that is mapped to b' ($\varphi(b) = b'$).

Homomorphisms preserve the source and target incidence relations between nodes and edges, but they are not required to preserve any other structure. Fibrations also preserve input sets, isomorphically; moreover, they collapse only nodes in the same equitable partition; that is, in synchronous fibers with color-isomorphic input sets. Thus Fig. 6.3b and Fig. 6.3c are

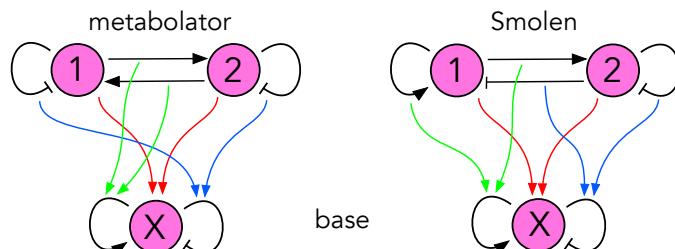


Fig. 6.2

Fibrations for the minimal equitable partition. The metabolator and Smolen networks.

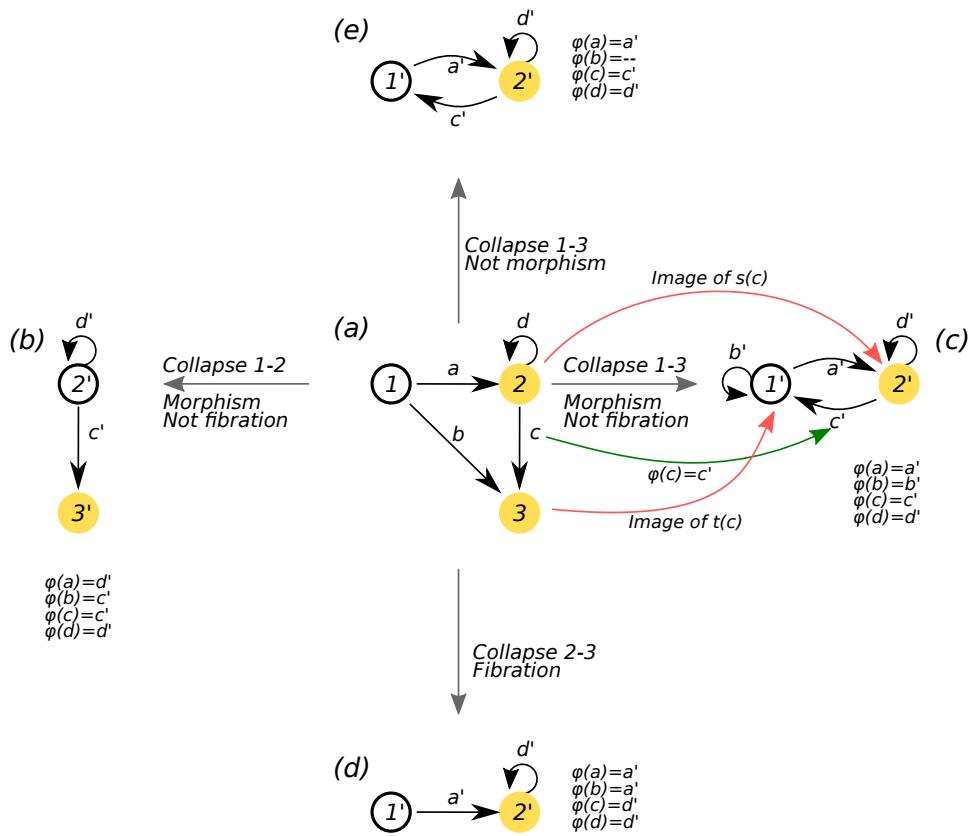


Fig. 6.3

Types of fibrations and homomorphisms in the FFF circuit. (a) The FFF circuit is the graph \$G\$ and we map this to four graphs \$H\$ as shown. The node component is described in this caption, whereas the edge component is described in the figure. Collapsing different nodes gives rise to different maps \$\varphi : G \rightarrow H\$: these can be either homomorphisms, fibrations or just maps. Maps contain homomorphisms, which contain fibrations. (b) Collapsing 1 and 2 (both mapped to node 2') with the edge map \$\varphi\$ shown in the figure provides a homomorphism but not a fibration. (c) Collapsing 1 and 3 (both mapped to node 1') with the edge map \$\varphi\$ shown is a homomorphism, but not a fibration because \$b'\$ cannot be lifted to any edge in \$G\$ (however, \$a'\$, \$c'\$ and \$d'\$ can be lifted to an edge in \$G\$). (d) Only when we collapse the fiber 2 and 3 (mapping them both to node 2') do we obtain a fibration. All edges in \$B\$ can be lifted to edges in \$G\$. (e) Collapsing 1 and 3 without mapping edge \$b\$ is not even a homomorphism. It is just a map among the node sets.

homomorphisms but not fibrations. The map of Fig. 6.3d, besides being a homomorphism, is quite special and different from the previous two. It is the only one of these examples that collapses only nodes in the same equitable partition. Therefore this homomorphism has special significance since the collapsed graph inherits the synchronous dynamics of \$G\$.

A homomorphism that collapses nodes in equitable partitions constitutes a graph fibration. Dynamical invariance is not satisfied by the other homomorphisms in Fig. 6.3, which are not fibrations.

In a more intuitive description, a graph fibration is a homomorphism in which every edge targeting an image node is an image of a *unique* edge, so multiple edges targeting the same node cannot be collapsed to fewer edges (as we did in Fig. 6.3b collapsing b and c into c'). Neither can new edges targeting the image of a node be added (as we did in Fig. 6.3c by creating b'). The only way to avoid these two conditions is to collapse equitable partitions. Thus, a fibration implies a stronger condition than the simple preservation of the incidence relation by homomorphisms. This extra conditions is captured by the lifting property, Section 6.5 below.

The definition of a fibration through the collapse of equitable nodes applies only to surjective fibrations. A more general definition of fibration, valid for all fibrations (surjective or not), uses the lifting property. We now elaborate on both definitions in turn.

6.4 Graph fibrations through equitable partitions

We now generalize these observations to give a general definition of a fibration, and discuss how fibrations lead to equitable partitions in this general context.

There are many equivalent ways to define graph fibrations. Normally they are defined using the ‘lifting property’ (see Section 6.5), but we can start by providing an alternative and possibly more intuitive definition based on input isomorphisms, which are local concepts.

Definition 6.6 Definition of graph fibration as a local input isomorphism. A graph homomorphism $\varphi : G \rightarrow B$ is a *graph fibration* if and only if, for all vertices $x \in V_G$, φ induces an isomorphism between the set of edges of G having x as target and the set of edges of B having $\varphi(x)$ as target. The graph G is the *total graph* of the fibration, B is called the *base graph*, and for every vertex x of B , the set of vertices of G that are mapped to x by φ is the *fiber (over x)*.

The condition says that, when looked at locally (i.e., at a specific vertex of the graph G and at its image in the graph B) the map provides an isomorphism between incoming edges. There is no condition on *outgoing* edges.

Based on this definition we can immediately state some properties that fibrations enjoy, relating them to equitable partitions.

Proposition 6.7 If $\varphi : G \rightarrow B$ is a fibration, its fibers define an equitable partition of G , the equitable partition *associated to φ* .

The converse of Proposition 6.7 is true:

Proposition 6.8 For every equitable partition of the nodes of G , there exists a graph B

and a surjective graph fibration $\varphi : G \rightarrow B$ such that two nodes are in the same cluster of the equitable partition if and only if they are mapped by φ to the same node of B .

In other words, the fibers of φ are exactly the clusters of the equitable partition, so fibrations are homomorphisms that collapse nodes belonging to the same cluster in an equitable partition.

Instead of starting with the fibration and considering its base, we can construct the base from the corresponding equitable partition.

Definition 6.9 Quotient network. Let G be a directed graph with an equitable partition P . The corresponding *quotient network* G/P is the graph whose nodes correspond to the elements of P (that is, the colors in the equitable partition). To construct the input edges to a given node we choose any node of that color and consider its input set. For each edge in the input set we define an edge in G/P by connecting nodes of the corresponding colors.

The map from G to G/P that assigns to each node of G its color in P is now a fibration, because each edge in G/P is the image of a unique edge in G ; that is, the lifting property holds. Moreover, the base of this fibration is G/P .

For instance, the identity map $id : G \rightarrow G$ (which maps each node and edge to itself) is a fibration, and the partition it induces (the identity partition) is equitable; in this case, no nodes are actually collapsed, though. More generally, every automorphism is, trivially, a fibration; hence all group symmetries are *a fortiori* equitable partitions. So fibrations generalize automorphism in a proper mathematical sense.

The notion of a fibration can easily be extended to graphs with various types of edges, such as a gene regulatory network which has both activation and repressor edges, or a brain network with excitatory and inhibitory synaptic connections. In this case, what we have to do is to restrict the notion of homomorphism to send nodes or edges of a certain type to nodes or edges of the same type.

Since nodes in the same equitable partition have isomorphic input trees, this is true also for nodes in the same fiber. More precisely:

Proposition 6.10 If $\varphi : G \rightarrow B$ is a fibration, and $\varphi(x) = \varphi(y)$, then T_x is isomorphic to T_y . Conversely, if x and y are two nodes of a graph G with isomorphic input trees (i.e., such that $T_x \sim T_y$), then there exists some graph B and fibration $\varphi : G \rightarrow B$ such that $\varphi(x) = \varphi(y)$.

As we have said in Section 5.8, the minimal equitable partition is the *coarsest* equitable partition that can be found. That is, it is the partition that collects all the symmetries that the network has, and partitions the network into the smallest number of fibers. Proposition 6.8 now implies:

Proposition 6.11 For every graph G , there is a graph \hat{G} , the *minimum base* of G and a surjective fibration $\mu : G \rightarrow \hat{G}$ whose associated fibers correspond to the minimal equitable partition. Because of the properties of the minimal equitable partition, $\mu(x) = \mu(y)$ if and

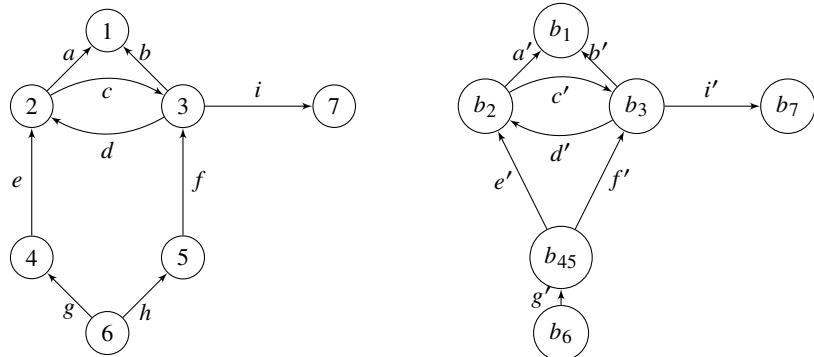


Fig. 6.4

Example of a fibration that is not minimal. The network G of Fig. 5.4 (left) and a nontrivial fibration $\varphi : G \rightarrow B$ to another network (right). Edges a, b, \dots are mapped to a', b', \dots respectively, with the exception of h , which is mapped to g' (like g); node i is mapped (injectively) to node b_i for all i , except for nodes 4 and 5, which are both mapped to b_{45} .

only if x and y have isomorphic input trees (i.e., $T_x \simeq T_y$). The fibration μ is the *minimal fibration* of G .

It turns out that this graph \hat{G} is unique (up to isomorphism), whereas the fibration $\mu : G \rightarrow \hat{G}$ is not necessarily unique, but it is unique on nodes.

Figure 6.5 shows a graph fibration that collapses the total graph G of Fig. 5.4 into the minimum base \hat{G} of G , and a minimal fibration $\mu : G \rightarrow \hat{B}$ that maps the total graph to the minimal base by collapsing all the fibers of the network.

To summarize this section, a graph fibration $G \rightarrow B$ maps the nodes of a graph G so that nodes that are mapped to the same node of B form an equitable partition. When we are looking only at surjective fibrations, every node in B has a preimage in G : therefore, except in trivial cases (i.e., when the map is a bijection), some distinct nodes in G are ‘collapsed’ into a single node. Nodes that are collapsed together by the fibration belong to the same *fiber*, and the size of the fiber is the number of collapsed nodes. Nodes that belong to the same fiber of surjective fibrations can synchronize their dynamics, which explains why fibrations are the natural way to describe synchronization.

All nodes in the same fiber have isomorphic input trees. Similarly, given an equitable partition of G , nodes in the same cluster have isomorphic input trees, and there is a surjective fibration $G \rightarrow B$ whose fibers precisely correspond to the clusters of the equitable partition.

The minimal surjective graph fibration is the fibration associated to the minimal (i.e., coarsest) equitable partition; in this case, two nodes $v_1, v_2 \in G$ belong to the same fiber \iff their input trees are isomorphic. \Rightarrow follows from the second part of Theorem 2 in (Boldi and Vigna, 2002a), and \Leftarrow is Proposition 23 in (Boldi and Vigna, 2002a). See also (Stewart, 2007). In other words, a surjective fibration is minimal if it collapses all nodes that can be collapsed.

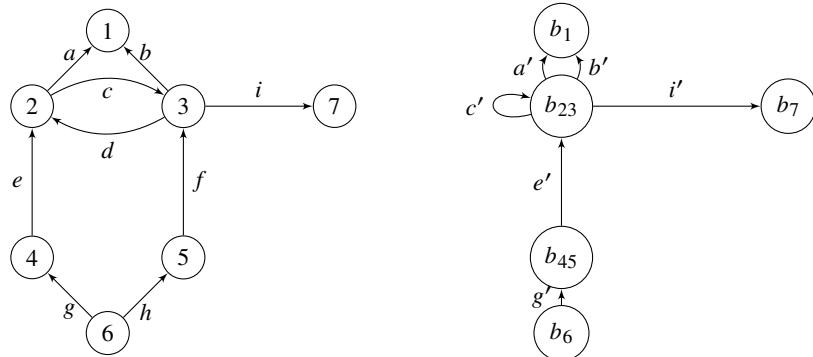


Fig. 6.5

Minimal fibration and minimal base. The network G of Fig. 5.4 (left) and its minimum base \hat{G} (right). A minimal fibration $\mu : G \rightarrow \hat{G}$ maps node i to node b_i except 4 and 5 (which are both mapped to b_{45}) and nodes 2 and 3 (which are both mapped to b_{23}). Edges a, b, \dots are mapped to a', b', \dots respectively, with the following exceptions: $d \mapsto c', f \mapsto e', h \mapsto g'$. The only other minimal fibration is exactly the same but maps $a \mapsto b'$ and $b \mapsto a'$.

6.5 The lifting property

For practical purposes, as well as to study fibrations in biological networks as in Chapters 14 and 15, Propositions 6.10 and 6.11 (specifying how fibrations and input-tree isomorphisms are related) are enough to understand fibration symmetries. That is, the search for fibration symmetries in real networks is performed by searching for the fibers of networks that correspond to the coarsest universal equitable partition or balanced coloring. The minimal base of the graph is then obtained by collapsing the fibers into a common node at the base. Algorithms to find the fibers of the network are based on finding the coarsest balanced coloring partition, and are discussed in Chapter 13.

In an alternative way (totally equivalent to Definition 6.6), graph fibrations can be defined independently using the so-called *lifting property*, which rigorously distinguishes a graph fibration from a generic homomorphism (Boldi and Vigna, 2002a):

Definition 6.12 Definition of graph fibration via the lifting property. A graph homomorphism $\varphi : G \rightarrow B$ is a *graph fibration* if and only if, for all edges $e' \in E_B$ and for all vertices $i \in N_G$, if $\varphi(i) = t_B(e')$ then there exists exactly one edge $e \in E_G$ such that $t_G(e) = i$ and $\varphi(e) = e'$. This unique edge e is the *lift* of e' at i .

In particular, every edge targeting $i' = \varphi(i)$ can be *uniquely* lifted to an edge in G targeting i . This is the *lifting property*.

The relation between Definition 6.6 and Definition 6.12 should be immediately clear: Definition 6.12 postulates the existence of an isomorphism between the edges entering into $\varphi(i)$ and the edges entering into i (existence and uniqueness, together, exactly define a

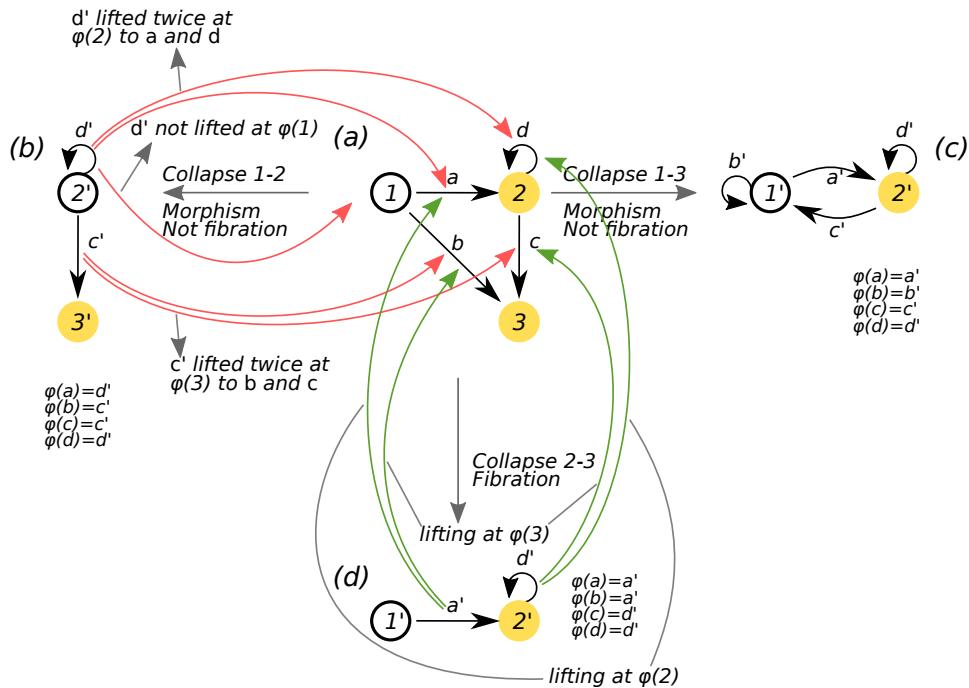


Fig. 6.6

The amazing world of the lifting property. Example of the lifting property for the fibration of Fig. 6.3d, and an example of too many lifts for the homomorphism of Fig. 6.3b and lack of a lift to node 1 in 6.3c.

one-to-one relation). In turn, this means that if $\varphi(i) = \varphi(j)$, then the set of edges entering into i and j must be isomorphic (because they are both isomorphic to the set of edges entering into $\varphi(i) = \varphi(j)$), which is what Definition 6.6 requires.

This definition means that multiple edges, targeting the same node in G , cannot be collapsed to fewer edges in B , unlike a homomorphism; neither can new edges targeting the image of a node be added. This is exemplified in Fig. 6.3d. This definition still allows for reduction of the network, because nodes with same number of inputs can be mapped to the same image.

The homomorphism of Fig. 6.3d satisfies the lifting property and hence it is a fibration, while the homomorphisms of Figs. 6.3b and 6.3c do not, so they are not fibrations.

To understand more deeply the lifting property, we examine why these homomorphisms fail to be fibrations. To do so, we test the lifting property. The fibration of Fig. 6.3d $\varphi : G \rightarrow B$ is defined by the following map between nodes: $\varphi(1) = 1'$, $\varphi(2) = 2'$, $\varphi(3) = 2'$, and by the map between edges: $\varphi(a) = a'$, $\varphi(b) = a'$, $\varphi(c) = d'$, and $\varphi(d) = d'$, as shown in the figure. We test lifting: for edge a' pointing to $2'$ in B , there are two nodes $x = 2$ and $x = 3$ in the fiber over $2'$. For each of these nodes in G we must check that there is a unique edge (the lift) pointing to them, so that upon fibration we get a' . Indeed, φ maps two edges to a' , namely a and b : one targets 2 and one targets 3. The same procedure

Fibration	Lifting Property	$i \in G$ *	$e' \in B$ **	$t_B(e')$	$\varphi(i)$	$e \in G$	$t_G(e)$ *	$\varphi(e)$ **
✓	a lift a' at 2	2	a'	2'	2'	a	2	a'
✓	d lift d' at 2	2	d'	2'	2'	d	2	d'
✓	b lift a' at 3	3	a'	2'	2'	b	3	a'
✓	c lift d' at 3	3	d'	2'	2'	c	3	d'

* These two columns should be the same for fibrations.

** These two columns should be the same for fibrations.

Table 6.1: Testing the lifting property for the fibration in Fig. 6.3d. All edges in B can be lifted uniquely, so the homomorphism is a fibration.

demonstrates the existence and uniqueness of the lifts of d' at 2 and 3 proving that φ is a fibration. The checks are shown in Table 6.5.

On the other hand, the homomorphism in Fig. 6.3b fails uniqueness of lifting for the edge d' at 2 (both a and d target 2 and are mapped to d'), and the same goes for an attempt at lifting c' at 3 (both b and c target 3 and are mapped to c'); moreover, d' cannot be lifted at 1. Again the tests are summarized in Table 6.5. Therefore the homomorphism is not a fibration. The failure of the lifting property in this homomorphism is somewhat subtle. Sometimes existence fails, sometimes uniqueness fails.

Finally, the homomorphism of Fig. 6.3c is also not a fibration as shown in Table 6.5. In principle, every edge in B can be lifted to an edge in G as indicated in the figure. However, there are lifts missing for node $x = 1$, so the homomorphism fails to be a fibration. This happens because these homomorphisms collapse two nodes (1 and 3) that are not input tree isomorphic: recall that a fibration can collapse only nodes with isomorphic input trees. Welcome to the world of fibrations, where these subtle differences are all relevant and important!

While the lifting property might look threatening, it is just another way to say that we cannot place any nodes with non-isomorphic input trees in the same fiber. For practical purposes, then, we first find the fibers using algorithms from Chapter 13, based on finding a balanced coloring, and then we apply the notion of fibration to reduce the network (i.e., to decide how *edges* can be mapped). Not the other way around, since we are not aware of any algorithm that can systematically find fibrations using the lifting property alone. Nevertheless, the lifting property is helpful when thinking in terms of a model of evolution of biological networks via duplication and speciation, discussed in Chapter 11.

A fibration can be surjective or not, injective or not (DeVille and Lerman, 2015b) (see Fig. 6.1). Surjectivity is mainly a technical requirement: we can make any homomorphism (and in particular, any fibration) $\varphi : G \rightarrow B$ surjective by modifying B , removing the nodes and edges that are not in the image of φ ; for this reason we focus mainly on surjective fibrations. On the other hand, we are mainly interested in non-injective fibrations, because those are the fibrations that collapse nodes and thus identify nontrivial clusters of synchronous nodes.

Example 11 We return to Example 2 and Fig. 4.2, which illustrates the key difference between graph homomorphisms and fibrations. Let G be the graph of Fig. 4.2a. The quotient

Fibration	Lifting Property	$i \in G$ *	$e' \in B$ **	$t_B(e')$	$\varphi(i)$	$e \in G$	$t_G(e)$ *	$\varphi(e)$ **
×	none	1	d'	2'	2'	none	none	none
×	> one	2	d'	2'	2'	a or d	2	d'
×	> one	3	c'	3'	3'	b or c	3	c'

* These two columns should be the same for fibrations.

** These two columns should be the same for fibrations.

Table 6.2: Testing the lifting property for the homomorphism in Fig. 6.3b. Edge d' cannot be lifted at 1, but it has two possible lifts at 2; edge c' also has two possible lifts at 3. Therefore the homomorphism is not a fibration.

Fibration	Lifting Property	$i \in G$ *	$e' \in B$ **	$t_B(e')$	$\varphi(i)$	$e \in G$	$t_G(e)$ *	$\varphi(e)$ **
×	none	1	b'	1'	1'	none	none	none
×	none	1	c'	1'	1'	none	none	none
✓	b lift b'	3	b'	1'	1'	b	3	b'
✓	c lift c'	3	c'	1'	1'	c	3	c'
✓	a lift a'	2	a'	2'	2'	a	2	a'
✓	d lift d'	2	d'	2'	2'	d	2	d'

* These two columns should be the same for fibrations.

** These two columns should be the same for fibrations.

Table 6.3: Testing the lifting property for the homomorphism in Fig. 6.3c. Neither b' nor c' can be lifted at 1, therefore the homomorphism is not a fibration.

in the sense of (Hahn and Tardif, 1997), that is, $G/\text{Aut}(G)$, is shown in Fig. 4.2b. The natural map from G to $G/\text{Aut}(G)$ maps each node of G to the node with the corresponding color in $G/\text{Aut}(G)$. This map is a graph homomorphism since it has at least an edge between collapsed nodes in the quotient if there was an edge between any nodes of the collapsed clusters in the original graph, thus preserving incidence under the collapse. However, it is not a fibration, because the edges (1, 5) and (4, 5) in Fig. 4.2a both map to the same edge (1, 2) in Fig. 4.2b.

In contrast, denote the equitable partition/coloring by P . Then the quotient network G/P is shown in Fig. 4.2c. Because the arrow from node 2 to node 3 is doubled, the lifting property holds. Now the map that assigns to each node of G the node of the corresponding color in Fig. 4.2c is a fibration. Input sets are preserved, and the admissible ODEs for the quotient network determine, precisely, the possible cluster dynamics for the coloring.

We also see that homomorphisms and fibrations have different effects on input trees. Homomorphisms can collapse them, but fibrations always preserve them.

Because the natural map from G to $G/\text{Aut}(G)$ is not a fibration but merely a graph homomorphism, input sets and input trees in the base are not isomorphic to input sets and input trees in the original graph. Since the map from G to G/P is a fibration, input trees

in the base *are* isomorphic to input sets and input trees in the original graph. Figure 6.7 illustrates this point.

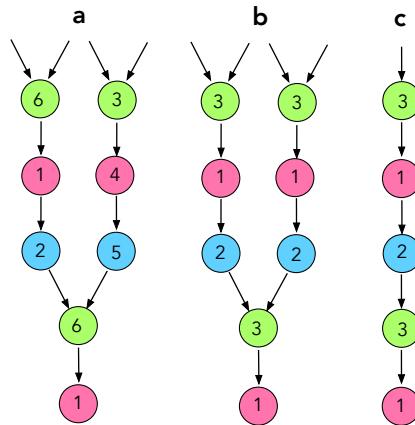


Fig. 6.7

Differences in input trees between homomorphisms and fibrations. (a) Input tree of node 1 in Fig. 4.2a. (b) Input tree of node 1 in Fig. 4.2c is isomorphic to that in (a), as it should be for a fibration. (c) Input tree of node 1 in Fig. 4.2b is not isomorphic to that in (a).

6.6 Hierarchy of symmetries from homomorphisms to fibrations (op-fibrations) to coverings to automorphisms

Having defined the main tool that we use in this book—fibrations—we revisit our initial purpose: to describe symmetry in graphs and synchronization in networks. The aim of this section is to summarize the two axes along which automorphisms and fibrations differ (local vs. global, in vs. in/out). To do this we introduce other local intermediate symmetries between these two (local+in = fibrations, local+out = op-fibrations, local+in+out = coverings) forming a hierarchy of symmetries from less to more strict: homomorphism → fibration (and op-fibration) → covering → automorphism.

As discussed, automorphisms capture a global type of symmetry: a bijective homomorphism from the structure to itself that preserves the *whole* structure. We can think of this as a *change of label* that does not change the shape. Applying this notion to graphs means that we consider graph homomorphisms $\varphi : G \rightarrow G$ that are bijections. In particular, such bijections preserve outgoing edges as well as incoming edges. Automorphisms (for graphs and in general) have a nice algebraic structure:

- They can be composed (as functions) and composition is *associative*, that is, $(\varphi_1 \circ \varphi_2) \circ \varphi_3 =$

$\varphi_1 \circ (\varphi_2 \circ \varphi_3)$: in other words, we can get rid of all parentheses when composing automorphisms.

- There is an *identity* map $1 : G \rightarrow G$, which maps each node/edge to itself. (In group theory the identity map is usually denoted by 1 , not *id*.) This map is an automorphism, and it is the identity for composition, because $\varphi \circ 1 = 1 \circ \varphi = \varphi$.
- Every automorphism is *invertible*, that is, for every automorphism φ there exists an automorphism φ^{-1} such that $\varphi \circ \varphi^{-1} = \varphi^{-1} \circ \varphi = 1$.

These three properties together make what mathematicians call a *group*; that is, the set $\text{Aut}(G)$ of all automorphisms of G is a group with respect to function composition. This nice, simple algebraic structure gives rise to a whole rich theory. There is a quotient graph $G/\text{Aut}(G)$ (Definition 4.9), but this is not suitable for describing synchrony patterns or the dynamics of clusters.

A key point, previously discussed, is that automorphisms are hardly the right way to describe symmetries of biology. The restrictions imposed by the global essence of an automorphism make it very unlikely that a real-world graph should possess any nontrivial automorphism (any automorphism other than the identity). So $G/\text{Aut}(G)$ is usually the same as G .

Fibrations overcome this obstacle. They offer another way to look at symmetry, with much humbler requirements. First, fibrations require that structure be preserved only locally and not globally. Second, they ask that only incoming edges are preserved, whereas outgoing edges are not taken into account. This is precisely what Definition 6.6 requires. All automorphisms are of course fibrations, but there are typically many more fibrations than automorphisms. Every fibration $\varphi : G \rightarrow B$ captures some of the fibration symmetry of the graph G . Proposition 6.10 tells us that the input trees are the same in every fiber; that is, $T_x \simeq T_y$ if $\varphi(x) = \varphi(y)$. This is important, because as long as we are interested only in the symmetry of vertices, we can simply look at their input trees. The analog of $G/\text{Aut}(G)$ for fibrations is the minimum base \hat{G} and the minimal fibration $\mu : G \rightarrow \hat{G}$. Again, looking only at the effect of μ on vertices, $\mu(x) = \mu(y)$ if and only if $T_x \simeq T_y$, as explained in Proposition 6.10.

6.6.1 Preserving outputs instead of inputs: op-fibrations

A brief conceptual stop is needed at this point. The duality global vs. local (automorphisms are global symmetries, fibrations are local symmetries) is clear. Less clear is why we look only at incoming edges and ignore outgoing edges. This is philosophically challenging.

The choice, here, comes from the meaning of directed edges in nature: a directed edge $x \rightarrow y$ represents a form of communication *from x to y*; the exact type of communication is not relevant, here (it will be different from case to case) but the idea is always that there is a flow of information going from one node (x) to another node (y). It is in the very nature of information that the recipient of a piece of information can change its state based on the information it obtained; the sender does not. This (trivial) observation explains why incoming edges are more important in practice than outgoing edges, and this is why fibrations are defined in that way.

One caveat: in some areas where network diagrams are commonly used, outputs can change the state of a node. The mass balance condition in biochemical networks imposes such a restriction, for instance. We can (and do) sidestep this issue by working with the ‘influence network’. This has extra *input* edges to a given node, which represent the effect of these outputs on the dynamics of that node. We have already discussed this point in subsection 3.3, but we repeat the message here. The component of an admissible ODE for node c has the form $\dot{x}_c = f_c(x_c, x_{i_1}, x_{i_2}, \dots)$ for a general map f_c and ‘input’ variables x_{i_1}, x_{i_2}, \dots . The equation specifies how node c *processes* the listed variables to affect the dynamics of its own variable x_c . In some areas of application, these variables may include outputs as well as inputs, but in the general theory we can represent their effects using the influence network, in which case we refer to them as inputs.

Of course, from a purely mathematical standpoint, we could consider a notion that is absolutely analogous to that of fibration, but where outgoing edges are considered instead and the output tree is preserved rather than the input tree. This notion is called *op-fibration* (‘op’ stands for opposite, and it is the standard categorical way to name things when you ‘reverse’ them). Now, op-fibrations are no different than fibrations: everything we know about fibrations applies *mutatis mutandis*. This time what counts is *output trees* (the trees of outgoing paths), \check{T}_x . We can imagine the op-fibration equivalent of Proposition 6.10: given an op-fibration $\varphi : G \rightarrow B$ and $\varphi(x) = \varphi(y)$, the output trees will be isomorphic, i.e., $\check{T}_x \simeq \check{T}_y$.

And again there will be a minimum op-fibration base \check{G} and a minimal op-fibration $\check{\mu} : G \rightarrow \check{G}$. Figure 6.13 shows an example of output trees.

Op-fibrations are not more informative or useful mathematically than fibrations: the two concepts are dual to each other. However, given that input trees are more relevant for dynamical systems in biology, we can put, in principle, op-fibrations in the cabinet of curiosities and leave them there. Except for a very important application: In chapter 26 we will show that op-fibrations and their output trees describe the symmetries in the learning process through backpropagation in deep neural networks, while fibrations and input trees describe the inference process.

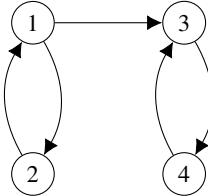
6.6.2 A graph having different fibrations and op-fibrations

To appreciate the difference between fibrations and op-fibrations, look at the graph G of Fig. 6.8. This graph has non-trivial fibrations and op-fibrations, but the two are different from each other.

To see why this fact happens, let us build the input trees and output trees of each node (see Fig. 6.10). As you can see, tree isomorphism clusters the nodes of the graph G in two different ways, depending on whether you are looking at input trees or output trees.

From the viewpoint of *inputs*, node 1 and 2 are totally indistinguishable because 1 only receives input from 2, and 2 only receives input from 1; 3 and 4, on the other hand, are totally different from each other, and also from 1 and 2, because 3 receives two inputs (it is in fact the only node in the graph with two incoming arcs), and 4 receive input from 3.

In fact, the minimum base \hat{G} over which G is fibered is shown in Fig. 6.10: nodes 1 and 2 belong to the same fiber, whereas 3 and 4 are in two different fibers.

**Fig. 6.8**

Fibrations and op-fibrations. This graph G has different non-trivial fibrations and op-fibrations. It has, however, neither non-trivial coverings nor non-trivial automorphisms.

Now, if we look at *outputs* instead the situation changes dramatically: this time it is 3 and 4 that are alike, because 3 only provides output to 4, and 4 only provides output to 3. Conversely, 1 is special in that it is the only node with two outgoing arcs, and 2 is special because it provides output to 1. The minimum op-fibration base \check{G} is shown in Fig. 6.11: this time, 3 and 4 belong to the same op-fiber, whereas 1 and 2 live in separate op-fibers.

6.6.3 Preserving input and output at the same time: coverings

The existence of two notions of local symmetry, one based on inputs and one based on outputs, raises a natural mathematical question: what if we want to preserve *both* input and output? Is there a natural notion that puts together the definition of fibration and that of op-fibration at the same time? more importantly: is it the same thing as an automorphism?

This notion exists, and it is called (*graph*) *covering* which is a symmetry of the input and output trees and applies to the learning and inference process in neural networks (Chapter 26).

A graph covering is a homomorphism $\varphi : G \rightarrow B$ such that, for every node x of G :

- φ induces a bijection between the edges of G incoming in x and the edges of B incoming in $\varphi(x)$ (like a fibration does);
- φ induces a bijection between the edges of G outgoing from x and the edges of B outgoing from $\varphi(x)$ (like an op-fibration does).

If we look again at the example of Fig. 6.8, while it has both non-trivial fibrations and non-trivial op-fibrations, it has no non-trivial coverings, simply because there do not exist two nodes that have at the same time isomorphic input trees and isomorphic output trees. In other words, the graph of Fig. 6.8 has no non-trivial coverings, and it is easy to see that this graph is also rigid, i.e., it has no non-trivial automorphisms.

Yet, it is possible to find graphs that have non-trivial coverings, but have no non-trivial automorphisms: Figure 6.12 shows a graph that covers the graph of Figure 6.8, but it is nevertheless rigid. This is because the notion of covering is *local*, albeit taking both inputs and outputs into account simultaneously.

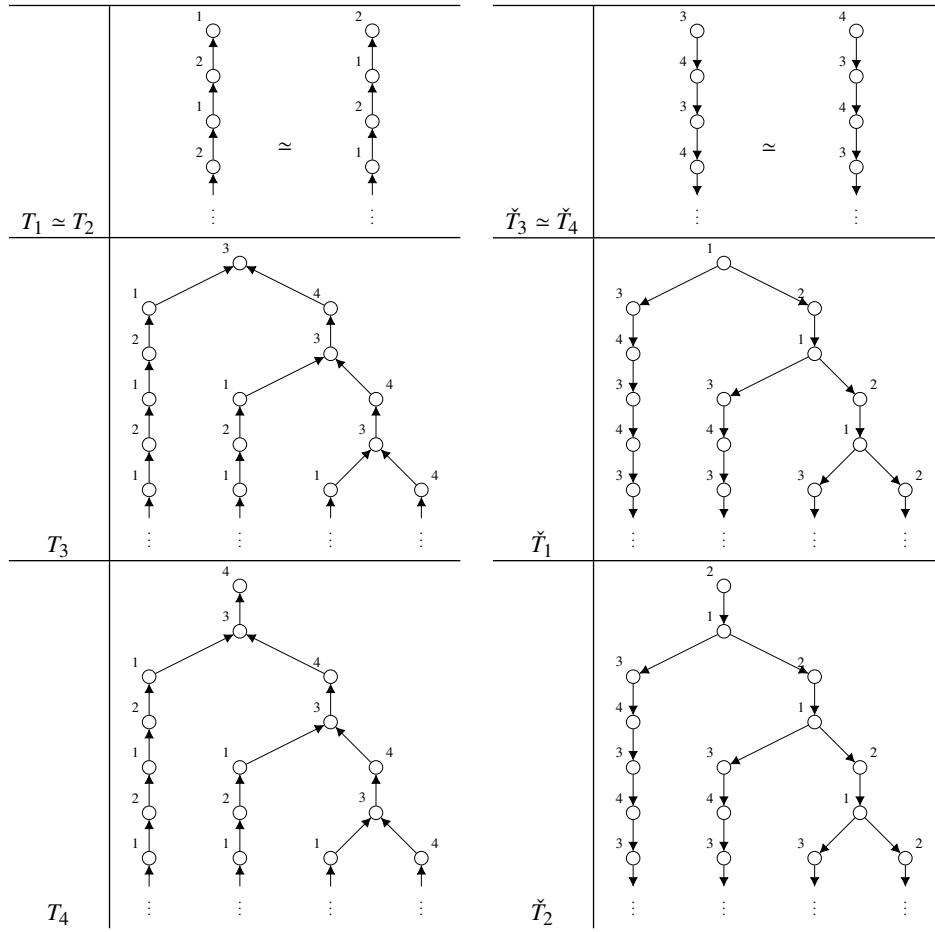
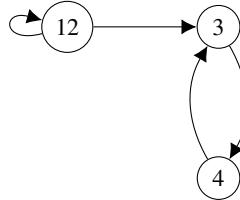


Fig. 6.9

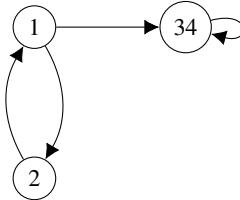
Input and output tree isomorphisms. The input trees T_i and the output trees \tilde{T}_i of all the nodes i of the network of Fig. 6.8, grouping those that are isomorphic.

6.6.4 Levels of symmetry

As we saw, a covering offers a more rigid kind of requirement than both fibrations and op-fibrations separately, but still far less rigid than that of an automorphism, because the bijection is required *locally* and not at a global level. This means that there are interesting examples of networks with different colorings according to the level of symmetry. In Fig. 6.13, we show a graph and the corresponding symmetries at the level of fibrations, covering and automorphisms. Figure 6.13 shows a network with three balanced colorings. One coloring arises from the strictest symmetry of automorphisms, preserving the whole structure (6 colors). This symmetry preserves the input and output of every node globally. An intermediate symmetry of the covering preserves the input and output trees of every node locally and produces 4 colors. It captures more symmetries than automorphisms, yet less than fibration. The fibration produces the highest symmetry with 3 colors in the base as

**Fig. 6.10**

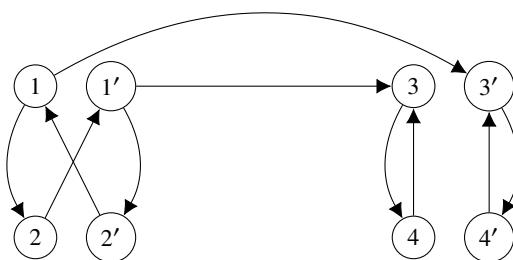
Minimum fibration base. The minimum fibration base \hat{G} on which the graph G of Fig. 6.8 is fibered; nodes 1 and 2 of G are both mapped to node 12 of \hat{G} , whereas $3 \mapsto 3$ and $4 \mapsto 4$.

**Fig. 6.11**

Minimum op-fibration base. The minimum op-fibration base \check{G} on which the graph G of Fig. 6.8 is op-fibered; nodes 3 and 4 of G are both mapped to node 34 of \check{G} , whereas $1 \mapsto 1$ and $2 \mapsto 2$.

shown. This intriguing example shows the hierarchy of symmetries at its best. The various types of graph symmetries are summarized in Table 6.4.

The level of automorphisms is unique in preserving global properties, whereas the other two levels preserve only certain local properties. This distinction between global and local symmetries is evident in other disciplines. In quantum field theory, a global symmetry implies that fields undergo transformations that remain constant across all space-time. An example is temporal invariance, which implies the conservation of energy. On the other hand, a local symmetry implies that fields can undergo local transformations that vary in space and time, such as gauge transformations (Jackson, 2012). For example, all

**Fig. 6.12**

Coverings. This graph covers the graph G of Fig. 6.8 mapping 1 and $1'$ to 1, 2 and $2'$ to 2 and so on.

	Local input isomorphism	Local output isomorphism	Global isomorphism
fibration	✓	✗	✗
op-fibration	✗	✓	✗
covering	✓	✓	✗
automorphism	✓	✓	✓

Table 6.4: Types of graph symmetries.

four fundamental interactions (i.e. gravity, electromagnetism, weak interaction, and strong interaction) are described by gauge local symmetries that keep the Lagrangians invariant (Berghofer et al., 2023), although the kind of locality that occurs in gauge theories is not analogous to the local condition of a fibration; a fibration is a less strict local condition than the gauge symmetries of the fiber bundle. More on this in Chapter 10.

6.7 Further thoughts on global versus local symmetries

Figure 5.3 shows that adding a single output edge to a network with global group symmetry can destroy that symmetry completely, even though the new edge does not affect the dynamics of any node in the original network. An extra output of this type preserves all existing fibration symmetries, although the new node forms an extra fiber.

This example suggests, correctly, that fibration symmetries are more robust than group symmetries to changes in the network topology, but in some ways it is also misleading, because it tends to suggest that the important difference is between inputs and outputs. Fibrations are defined in terms of input isomorphisms; outputs as such are irrelevant in this definition. (Outputs do affect the dynamics, but they are represented in the theory as inputs to the nodes that are influenced by their signals.) In contrast, automorphisms and coverings also preserve output sets (defined in the obvious way). Therefore the definition of an automorphism and coverings requires the preservation of structure that is irrelevant to synchronization.

It is tempting to trace the destruction of symmetry to this difference between inputs and outputs. However, the actual situation is subtler. For a start, if the extra edge in Fig. 5.3 is reversed, this *also* destroys the group automorphism, as in Fig. 6.14. In this case, the new edge also destroys *part of* the fibration symmetry, by breaking up the fiber {2, 3}. However, the other nontrivial fiber {4, 5} remains untouched.

What this example tells us is that the most important distinction between automorphisms and fibrations is that the former are global properties of the network, while the latter are local. Any change anywhere in the network, be it a new input or a new output, can therefore affect the automorphism group, often in a fairly drastic way. (It is easy to concoct examples where an extra edge actually *increases* the size of the automorphism group. Indeed, this

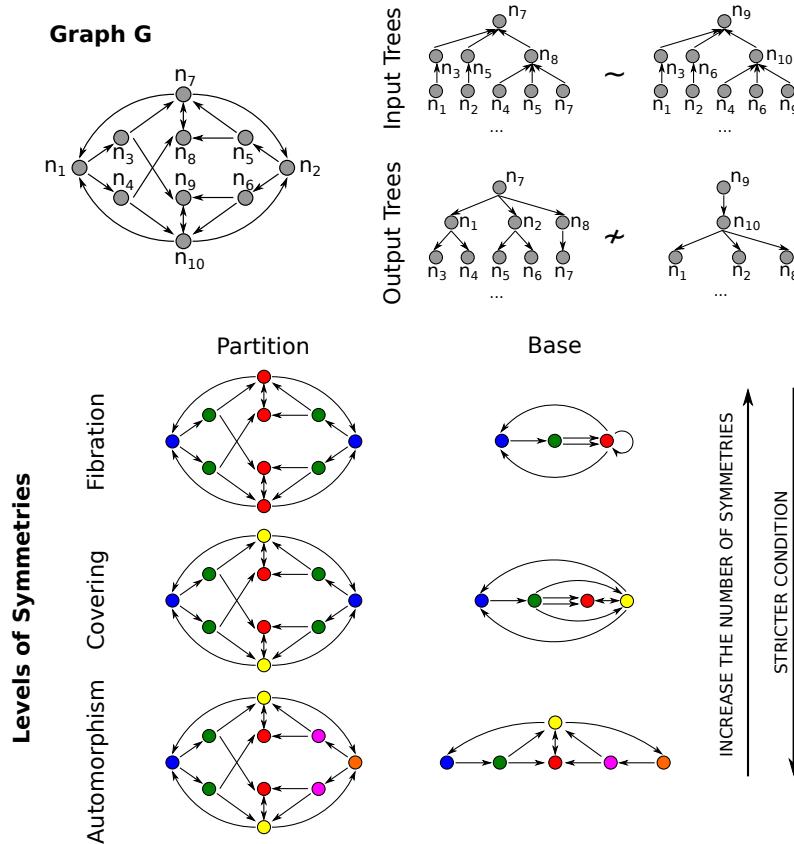


Fig. 6.13

Level of symmetries of a graph. For each level of symmetry (i.e., fibrations, coverings, automorphisms), there exists a partition of the nodes of a graph G induced by the definition of the level. The original graph can be compressed into a smaller graph (its *base* B). In the *base* B , nodes represent equivalence classes (indicated with colors).

happens to Fig. 23.3b when it is idealized to Fig. 23.3d. However, the most likely effect of a change is to make the automorphism group smaller.)

For fibration symmetries the distinction between inputs and outputs has more relevance. New outputs do not change the input set of the node impacted by the addition, so such a fiber remains unchanged. What does change is the fiber of the node to which the output goes, since it is an input to that node.

If a new edge is added, it changes the input set of its target node, splitting that node off from its previous fiber. All other input sets remain unchanged, so the symmetry becomes smaller, but much of it usually survives. Ironically, it is possible for the node that is split off in this manner to end up in some other fiber for a suitable fibration.

Despite all of the above caveats, two basic statements remain largely valid:

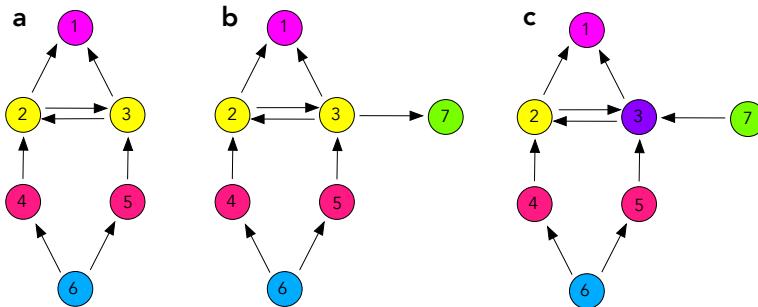


Fig. 6.14

Global versus local. (a) The network of Fig. 5.3 with its minimal balanced coloring, defined by the automorphism group. (b) The effect of adding a new output edge and target node 7. There are now no automorphisms except the identity. The fibration symmetries remain and the coloring remains balanced (with a new color for node 7). (c) The effect of adding a new input edge and source node 7. There are now no automorphisms except the identity. The fibration-induced synchrony of nodes 2 and 3 is destroyed because node 3 now has an extra input. However, the fibration-induced synchrony of nodes 4 and 5 persists.

- Fibration symmetries are generally more robust to small changes in network topology than automorphism (group) symmetries.
- Changes to the output edges of a node have no effect on its input set. Changes to the input edges of course do. Which fibers survive such a change depends on where the outputs go.

6.8 Example of fibrations, synchrony, colorings, and balance

We saw in Chapter 6 that graph symmetry theory can be presented in two distinct ways, which are mathematically equivalent (DeVille and Lerman, 2015b,a) but emphasize different aspects of the structure: balanced colorings and graph fibrations. The two are linked by the concept of an admissible ODE—model equations that reflect the network topology—and quotient networks, a construction that collapses synchronous nodes together, into a single node in a collapsed network that is analogous to the base of the fibration. Fibrations focus on the map between the nodes of the network and those of the quotient network; balanced colorings focus on the clustering of nodes induced by identifying those that are synchronous. The quotient network focuses on the interactions between the clusters; admissible ODEs focus on the consequent dynamics and synchronization.

Below, we exemplify these ideas using two common networks from biology and two modified networks whose mathematical features shed light on the general theory.

We begin with two networks from (Leifer et al., 2020, Fig. 4), there called replica and

base Fibonacci circuits, shown respectively in Fig. 6.15a and b. All nodes have the same type, and also all arrows have the same type (repressor/inhibitory). These two networks yield two distinct classes of models. Network Fig. 6.15a has symmetry fibration (which is also a group \mathbb{Z}_2 , which swaps nodes 1 and 2, and also swaps 3 and 4). Network Fig. 6.15b has trivial group symmetry (i.e. the identity): in particular, node 5 has one input but node 6 has two.

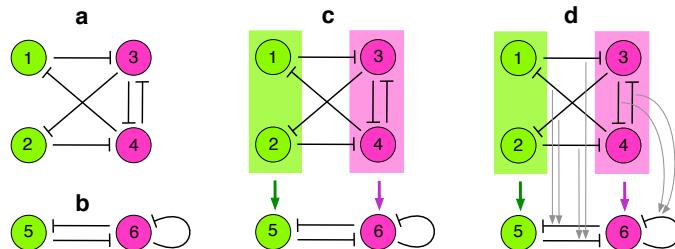


Fig. 6.15

A quotient network and the corresponding fibration. (a) Replica Fibonacci circuit with nodes classified into two disjoint subsets (colors). (b) Base Fibonacci circuit. (c) Fibration map on nodes identifies all nodes of the same color; the base is the quotient network of the replica for the coloring shown. (d) Fibration map on arrows maps each arrow in the replica to the arrow in the base with the same colors for its head and tail nodes.

Nodes 1 and 2, colored green, each receive one input arrow. Nodes 3 and 4, colored red, each receive two input arrows. Synchrony in particular requires nodes to have the same number of input arrows (of each type); that is, the same input tree to level 1, up to isomorphism. Therefore nodes 1, 2 cannot synchronize with nodes 3, 4. However, it might be possible for nodes 1 and 2 to synchronize, and/or for nodes 3 and 4 to synchronize.

If nodes 1 and 2 synchronize then so must their inputs, which implies that nodes 3 and 4 must also synchronize with each other (though not necessarily with nodes 1 and 2). Then the inputs to nodes 3 and 4 must synchronize (because in this case the only inputs are from a single node, namely nodes 4 and 3 respectively). Thus we can work backwards along the chain of inputs, which is why synchronization can be described using input trees.

Since synchrony requires nodes to have isomorphic input trees to all levels, we could use this condition as a test for possible synchrony patterns. Alternatively, we can take a short cut using the colors. In Fig. 6.15a each green node receives one input from a red node, and no other inputs. Similarly, each red node receives two inputs, both from a green node, and no other inputs. Thus the coupling between nodes preserves their colors. That is, it is a balanced coloring as in Definition 5.7. As we explained above, this condition ensures the existence of synchronous states in the dynamics.

6.9 Fibers and minimal balanced colorings

The fiber terminology of Section 6.4 arises because there is an alternative way to describe the quotient network construction, illustrated in Fig. 6.15cd, which is the graph fibration introduced in Section 6.2.

A graph fibration is a map from network **(a)** to network **(b)** that preserves certain aspects of the graph structure (Definition 6.12). More precisely, it is a pair of maps: one acting on nodes, the other on arrows. In **(c)** we show the map acting on nodes (thick colored arrows). Nodes 1 and 2 both map to node 5; nodes 3 and 4 map to node 6. In other words, the fibration preserves the colors of the nodes. The map on arrows is depicted by the gray arrows in **(d)**. It preserves the colors of head and tail nodes; for example the arrow from node 4 to node 1 has red head and green tail, so it maps to the arrow from 6 to 5.

In order to be a fibration, such a (pair of) map(s) must also preserve the input sets of nodes (level 1 of the input tree). For example, nodes 1 and 2 both have one input arrow; so does node 5; moreover, the sets of input arrows in **(a)** and **(b)** correspond under the fibration. Similarly nodes 3, 4, and 6 have two input arrows, and again they correspond under the fibration.

This final condition is the analog of balance from the point of view of fibrations. More precisely, given a coloring—balanced or not—we can define a new network by identifying all nodes of a given color. We can also map nodes of the original network to the corresponding node of the new one, by mapping a node to its color. However, this map may not extend consistently to a map of arrows, so the new network does not have a well-defined topology. If a consistent map of arrows exists, we have a homomorphism, but this need not preserve input sets. However, if the coloring is balanced, the map of the nodes does extend consistently to a map of arrows *and* how they are arranged in input sets. Now we obtain a well-defined quotient network, and the maps of nodes and arrows form a fibration.

Passing from the fibration viewpoint to the balance/quotient viewpoint is straightforward. Given a fibration, the fiber of a node in the image network is the set of all nodes that map to it. Then we assign a separate color to each fiber. The fibration condition implies that this coloring is balanced, and the image is the quotient network for that coloring. Conversely, given a balanced coloring, there is a natural map from the original network to the quotient. This maps each node to its color (thought of as a node in the quotient), and maps arrows to preserve the input set and the colors, so it determines a fibration.

Figure 6.15c is in fact the *minimal fibration* or *coarsest balanced coloring* as in Definition 5.6. That is, the number of synchrony classes (clusters, fibers, colors) is as small as possible. Every network has a unique minimal fibration (Stewart, 2007).

We now modify Fig. 6.15a. We do not claim that the resulting network arises naturally in biology: it is introduced to illustrate some important mathematical features. Figure 6.16a rewrites the arrows of Fig. 6.15a, but retains the numbers of input arrows and the colors of their heads and tails. Accordingly, the quotient network **(b)** is the same as before. There is a corresponding fibration, shown in **(c)** for nodes only. Again this is the minimal fibration

(we cannot make node 1 red as well, while retaining balance, because it has only one input arrow).

However, now there is another balanced coloring, hence a corresponding fibration, shown in (d). This fibration is not minimal. It has three colors, with a new color blue for node 2. Unlike the Fibonacci network, this coloring is balanced, because both red nodes receive inputs from node 1. The quotient network, with three nodes, is also shown in (d). Thus, although the minimal fibration is unique, there can be other fibrations, hence other balanced colorings, that use more colors.

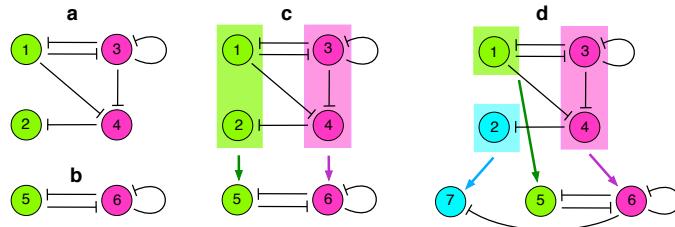


Fig. 6.16

Example of fibration. (a) A rewiring of the replica Fibonacci circuit, with nodes classified into the same two colors. (b) Fibonacci circuit, base, unchanged. (c) Fibration on nodes identifies all nodes of the same color. Arrows are mapped (map not shown) according to their head and tail colors as before; the quotient network of network (a) for this coloring is network (b). (a) A different fibration of (a) with three colors, leading to a quotient network with three nodes.

6.10 Beyond the minimal balanced coloring

A network may have many different colorings, in addition to its minimal coloring. The set of all possible balanced colorings has its own structure as a partially ordered set. Indeed, in (Golubitsky and Stewart, 2023, Chapter 13) and Aldis (2010); Stewart (2007) it is proved that the set of all colorings of a network forms a *lattice*, in the sense of a partially ordered set that satisfies certain axioms; see (Davey and Priestley, 2010). The set of all balanced colorings is a sublattice of this.

The partial order is refinement:

Definition 6.13 A coloring κ_1 is *finer* than, or *refines*, a coloring κ_2 if $\kappa_1(c) = \kappa_1(d)$ implies $\kappa_2(c) = \kappa_2(d)$, for all pairs of nodes c, d from the set C of nodes. We denote this relation by $\kappa_1 \leq \kappa_2$. The coloring κ_1 is *coarser* than κ_2 if κ_2 is finer than κ_1 . We denote this by $\kappa_1 \geq \kappa_2$.

A finer coloring has more colors, and these are obtained by subdividing color classes of the coarser coloring. Conversely, a coarser coloring arises by amalgamating color classes of a finer one. The finest coloring occurs when all nodes have different colors; trivially this

coloring is balanced. The coarsest coloring is the 1-coloring in which all nodes have the same color. It is balanced if and only if the network is homogeneous; that is, all nodes are input isomorphic.

Several algorithms exist to determine some or all balanced colorings of a given network: see Chapter 13.

Example 12 Consider the 5-node network of Fig. 6.17 (left). Assume all nodes have the same node-type with state space \mathbb{R}^k , and all arrows have the same arrow-type.

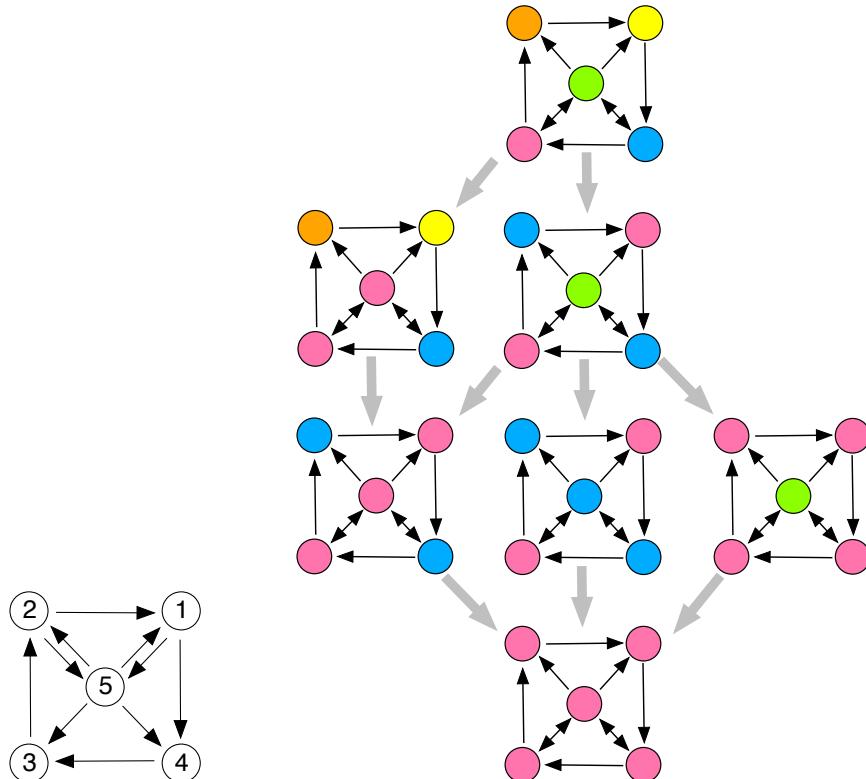


Fig. 6.17

Lattice of balanced colorings. *Left:* A five-node network with seven balanced colorings and a minimal fibration which fibers to a single node with two loops. *Right:* Lattice of balanced colorings for this 5-node network. Grey arrows represent coarsening of the coloring (head is coarser than tail). Compositions of grey arrows not shown. Each grey arrow also corresponds to a fibration in which the coloring becomes coarser. Figure 6.18 shows the corresponding bases.

The algorithm of Kamei and Cock (2013a) is able to calculate all balanced colorings of a graph and confirms that the network of Fig. 6.17 has exactly 7 balanced colorings. The colorings and the lattice that they form under refinement are shown in Fig. 6.17 (right).

The lattice of balanced colorings can also be considered in terms of fibrations. The

relation of coarsening, which is the reverse of refinement, corresponds to a fibration: the coarser coloring fibers over the finer one. The bases of the fibrations are shown in Fig. 6.18.

Some of the bases in this figure also fiber over other bases with fewer colors. For example, they all fiber over the minimal base at the lower right.

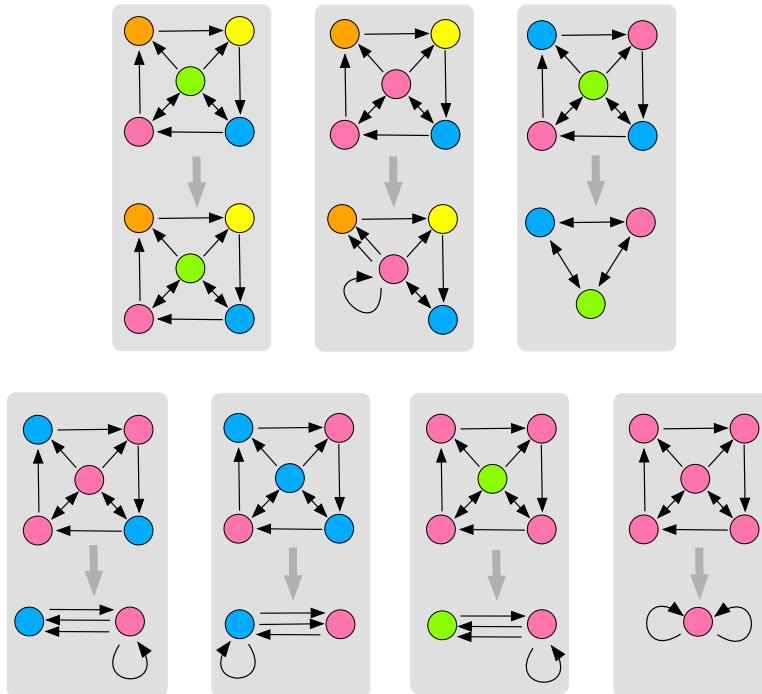


Fig. 6.18

Bases for the fibrations corresponding to the balanced colorings of the 5-node network in Fig. 6.17 (right). Each grey arrow represents the fibration that maps nodes of a given color in the network to the node of the same color in the base. The associated fiber is the set of nodes of that color; that is, the corresponding cluster. Multiple edges and self-loops are common.

The graph fibration formalism for synchronization and the associated balanced coloring and fibers can be cast into the groupoid formalism developed in (Stewart et al., 2003; Golubitsky et al., 2005a; Golubitsky and Stewart, 2006, 2016, 2023). This chapter provides a brief, informal introduction to the main concepts of the formal theory of groupoids in networks described in the cited sources. We relate the groupoid framework to the fibration and balanced coloring frameworks. The discussion is restricted to basic topics that are relevant to cluster synchrony. This chapter need not be read in detail.

7.1 Formulation in terms of symmetry groupoids

In this chapter we reinterpret parts of Chapter 6 from a different but closely related point of view, emphasizing algebraic structure alongside graph topology. We reformulate the problem of finding balanced colorings in a theoretical manner using an algebraic tool called the *symmetry groupoid*. We describe this concept in detail in the next section, after recalling some background.

Every balanced coloring induces a local isomorphism between nodes with the same color: this local isomorphism is, concretely, a bijection between the input sets of the two nodes, i.e., between the sets of their incoming edges. The symmetry groupoid is a formal way to represent a collection of input isomorphisms: every element of the groupoid contains the bijections between input sets of all pairs of nodes with the same indegree. Bijections are functions, so when they can be composed and their composition is associative. Moreover, any bijection has an inverse, and here this is also an input isomorphism. However, input isomorphisms cannot always be composed, because each bijection is local: its domain is the input set of a node, and its codomain is the input set of another node. Thus the natural algebraic structure of the set of all input isomorphisms—the ‘local symmetries’ of the network—is that of a groupoid. Informally, we can think of this as a group whose composition operation is defined only partially.

In marked contrast to the automorphism group (which, as we said, contains only one element), the symmetry groupoid of the Frucht graph contains 864 elements, see Example 6. It has one element for every possible isomorphism between incoming edges of nodes with the same indegree, and in the Frucht graph all nodes have the same indegree, which is 3. The symmetry groupoid consists of all bijections between each pair of input sets. There are 12 nodes, so there are $12 \times 12 = 144$ pairs of nodes; for each pair of nodes (c, d) , the input sets of c and d contain three edges each, so we can choose $3! = 6$ possible bijections,

therefore we have $144 \times 6 = 864$ input isomorphisms. Every balanced coloring induces a set of input isomorphisms, but not all sets of input isomorphisms correspond to balanced colorings.

As already remarked, the algorithm of Kamei and Cock (2013a) shows that the Frucht graph has exactly five balanced colorings, shown in Fig. 5.21. In coloring (a), all nodes have the same color; this coloring is balanced because all nodes have the same in-degree: it represents complete synchrony. Coloring (e) is trivial: all nodes have different colors; here no pairs of nodes synchronize. Coloring (b) and (c) are more interesting: they both have two colors, and represents a state in which nodes group into two synchronized clusters: in one case (Fig. 5.21b), $\{1, 3, 4, 5, 6, 7, 9, 11, 12\}$ and $\{2, 8, 10\}$; in the other case (Fig. 5.21c) $\{1, 2, 3, 6, 7, 9, 10, 12\}$ and $\{4, 5, 7, 11\}$. In both cases, the two clusters are the fibers of the fibration obtained by mapping each node to its color; Fig. 7.1 shows the fibration for Fig. 5.21b. Edges are mapped according to their source and target colors.

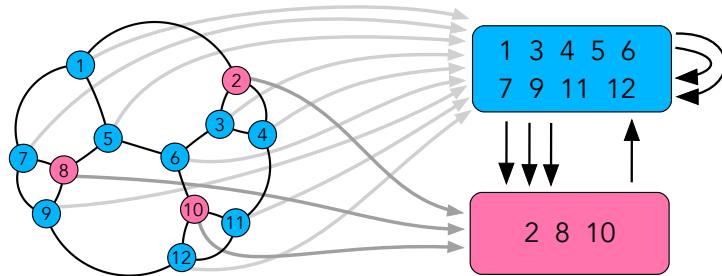


Fig. 7.1

Fibration (gray arrows) corresponding to balanced coloring (b) of the Frucht graph. Edges of main graph are bidirectional; those of quotient are directed, shown by black arrows.

Finally, coloring (d) uses three colors and is perhaps the most surprising one: in this final case, there are three clusters of the same size (four nodes in each cluster). Observe that (d) is a refinement of (c): nodes that are in the same cluster with respect to (d) are also in the same cluster with respect to (c); more precisely, the blue cluster of (c) (with 8 nodes) has split into two equally sized clusters.

As observed in (Stewart, 2007; Kamei and Cock, 2013a), the balanced colorings form a lattice according to refinement, see Section 6.10.

7.2 Groupoids

As we mentioned, although the input isomorphisms of a network seldom form a group, they do have a lot of algebraic structure. Indeed, their natural structure is that of a ‘groupoid’. Groupoids are of more recent vintage than groups, and were introduced by (Brandt, 1927). They resemble groups, except that sometimes two elements cannot be composed. For completeness, we provide the formal definition of a groupoid in Section 7.8; however, for

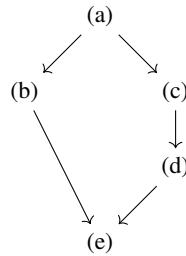


Fig. 7.2

The lattice of balanced colorings of the Frucht graph. Every arrow represents a refinement (head is finer than tail). The letters refer to Fig. 5.21.

present purposes, examples suffice to convey the key ideas. This section also shows that the groupoid formalism is of some use in the study of network synchronization, but its role is mainly to provide an abstract algebraic setting analogous to the way groups codify automorphisms. For the moment, we content ourselves with one example: Fig. 2.1a. This network has precisely five input isomorphisms. Expressed as bijections of input sets (which consist of all edges entering in a node). These bijections are:

$$\begin{aligned}\varepsilon_1 &= \begin{pmatrix} e_2 \\ e_2 \end{pmatrix} & \varepsilon_2 &= \begin{pmatrix} e_1 & e_4 \\ e_1 & e_4 \end{pmatrix} & \varepsilon_3 &= \begin{pmatrix} e_3 & e_5 \\ e_3 & e_5 \end{pmatrix} \\ \tau &= \begin{pmatrix} e_1 & e_4 \\ e_3 & e_5 \end{pmatrix} & \tau^{-1} &= \begin{pmatrix} e_3 & e_5 \\ e_1 & e_4 \end{pmatrix}\end{aligned}$$

Here the notation for the bijections is a variant of the usual notation for permutations. The top row shows elements of the domain; the bottom row shows their images in the codomain.

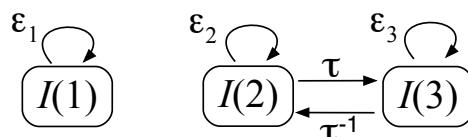


Fig. 7.3

Symmetry groupoid. Domains and codomains of the five bijections that form the symmetry groupoid of the network in Fig. 2.1a.

Input set $I(1)$ is disconnected from the other two in this figure. This happens because it is not input isomorphic to $I(2)$ or $I(3)$ (node 1 has only one incoming edge, whereas both 2 and 3 have two incoming edges). The maps $\varepsilon_1, \varepsilon_2, \varepsilon_3$ are distinct ‘identity maps’, because when they can legally compose with any other map, the result is that map. For example, $\varepsilon_3\tau = \tau$. We also have $\tau\tau^{-1} = \varepsilon_3$ and $\tau^{-1}\tau = \varepsilon_2$. On the other hand, the composition $\varepsilon_2\varepsilon_3$ is not defined because the codomain of ε_3 is $I(3)$, and this is not the domain $I(2)$ of ε_2 . We consider only isomorphisms that respect edge type (e.g., we cannot map a repressor to an activator, or the other way round).

We can calculate the ‘multiplication table’ of this groupoid, Table 7.1. Here the entry

Table 7.1: Multiplication table of the groupoid.

	ε_1	ε_2	ε_3	τ	τ^{-1}
ε_1	ε_1	—	—	—	—
ε_2	—	ε_2	—	—	τ^{-1}
ε_3	—	—	ε_3	τ	—
τ	—	τ	—	—	ε_3
τ^{-1}	—	—	τ^{-1}	ε_2	—

in row α and column β is the composition $\alpha\beta$, which is obtained by applying first β , then α . Diagrams like Fig. 7.3 or tables like Table 7.1 are useful when contemplating groupoid structure. But even then, groupoids are something of an acquired taste.

7.3 Input isomorphisms between input sets

As shown in Section 4.2, a group symmetry gives rise to an orbital partition (Definition 4.6) which we can show is also a balanced coloring, the orbit coloring. An orbit coloring is always balanced. However, a balanced coloring may not necessarily represent an orbit (Section 4.2). More generally, every balanced coloring is determined by some fibration. Now the colors are given by the fibers of the fibration, and balance is ensured by the definition of a fibration.

Any orbit coloring, being balanced, corresponds to a fibration in which nodes and arrows map to their orbits. However, not all fibrations arise in this manner, leading to the generalization from groups to groupoids to be discussed in this chapter. The role of groupoids is theoretical, and in practice somewhat peripheral; in contrast, the concept of balance/fibration is essential.

We have seen so far that cluster synchronization is captured by the existence of a symmetry fibration emerging from isomorphic input trees. The input trees need not necessarily be colored to be isomorphic. The theory of groupoids is based on the input sets rather than on input trees; the symmetry groupoid \mathcal{B} of the network is the set of all *input isomorphisms* between its input sets, with natural composition operations. For these isomorphisms to be relevant to cluster synchronization, the input sets are given a balanced coloring, and the isomorphisms are required to respect this coloring. These ‘color-preserving input isomorphisms’ form the ‘symmetry groupoid’ of the coloring, and this is a subgroupoid of \mathcal{B} . In this chapter we formalize this notion and compare it with symmetry fibrations and symmetry groups.

Since input isomorphisms are bijections between input sets, to lay out the formal definition of symmetry groupoid we first recall the definition of the input set of a node (Definition 5.1) and then proceed introducing the notion of an ‘input isomorphism’.

According to Definition 5.1, the input set of a node consists of the node itself, together with its incoming *edges*. In multigraphs (i.e., graphs that allow for parallel edges) the

node may possess different edges coming from the same source. Such ‘parallel’ edges are uncommon in biological networks, though they sometimes occur. To keep the discussion (and notation) simple, we assume that the network under consideration has no multiple edges: in that case, we can identify an edge with the pair (j, i) of nodes, where i is the target node and j the source node. (Some authors use the reverse convention, replacing (j, i) by (i, j) .)

Furthermore, when discussing the input set of node i , we can identify the input edge with its source node j . We make this simplification in this section. It assumes that there are no multiple edges between the same pair of nodes, and no self-loops. Without this simplifying assumption, we must also label the edges with their own symbols, complicating the notation without improving understanding.

Remark The general theory of (Golubitsky and Stewart, 2023) permits these features; indeed, they are mathematically necessary because even when a network has no such features, they can arise in a quotient network by a balanced coloring; equivalently, in the base of a fibration. The lifting property—restricted ODEs on the synchrony space are *precisely* the admissible ODEs for the quotient network—works only when networks with multiple edges and self-loops are permitted. Self-loops are common in biology (as ‘autoregulation’ in genetic networks, for instance). Multiple edges are less common, in part because they can often be considered as weighted edges with larger weights.

Given a node i , we denote its input set by:

$$\text{In}(i) = \{i : \text{input}_1, \dots, \text{input}_{k_{\text{in}}}\}.$$

(Alternative notations are ∂_i^{in} and $I(i)$.) This notation emphasizes that there is a ‘base’ node i with in-degree k_{in} that receives input edges from nodes $\text{input}_1, \dots, \text{input}_{k_{\text{in}}}$. For example, the input sets of nodes in the network of Fig. 7.4a are:

$$\begin{aligned} \text{In}(1) &= \{1 : 2, 3\} & \text{In}(2) &= \{2 : 3, 4\} \\ \text{In}(3) &= \{3 : 3, 5\} & \text{In}(4) &= \{4 : 6\} \\ \text{In}(5) &= \{5 : 6\} & \text{In}(6) &= \{6 :\} \\ \text{In}(7) &= \{7 : 3\}, \end{aligned} \tag{7.1}$$

and are depicted in Fig. 7.4b.

Definition 7.1 **Input isomorphism.** An *input isomorphism* between nodes i and j is a bijective map

$$\beta : \text{In}(i) \rightarrow \text{In}(j)$$

If such a map exists, the input sets are *input isomorphic*:

$$\text{In}(i) \simeq \text{In}(j). \tag{7.2}$$

For example, an input isomorphism between $\text{In}(1)$ and $\text{In}(2)$ in Fig. 7.4b is given by the

map $\tau_{1 \rightarrow 2} : \text{In}(1) \rightarrow \text{In}(2)$, defined as follows:

$$\tau_{1 \rightarrow 2} = \begin{pmatrix} 1 & 2 & 3 \\ \downarrow & \downarrow & \downarrow \\ 2 & 3 & 4 \end{pmatrix}, \quad (7.3)$$

which maps base node 1 to base node 2, and their respective inputs: 2 to 3, and 3 to 4 (see Fig. 7.4a). The only condition for this map to exist is that it should be bijective, that is, possess a well-defined inverse. In practical terms, the only condition for an input isomorphism in a given network is that it maps two input sets that have the same in-degree k_{in} .

Remark In the more general setting of (Golubitsky and Stewart, 2023, Sections 7.6, 9.2), where edges can have different types, an input isomorphism must also preserve the types of edges and of the base nodes. That is, if $e \in \text{In}(i)$ then $\beta(e) \in \text{In}(j)$ must have the same type as e . Now the interpretation is that nodes i and j have the same number of input edges for each type. Moreover, multiple arrows (having the same target and the same source, but not necessarily the same type) are not only permitted, but necessary, in order for possible synchronous dynamics to be determined precisely by the admissible ODEs for the quotient network.

The example network of Fig. 7.4a has in total 28 input isomorphisms, which can be obtained by inspection. These constitute its symmetry groupoid. We saw that the symmetry group defines orbits, which give balanced colorings, or equivalently fibrations, but not all colorings are balanced. Analogously, the symmetry groupoid defines colorings via fibrations, and these are balanced, but other colorings need not be.

Specifically, two input sets may be input isomorphic, yet fail to synchronize. The relation of input isomorphism is in general not balanced. This happens because the input set captures only the first layer of the input tree. To transform this truncated input set into a carrier of a fibration symmetry, we must color it with a balanced coloring. Then, to capture the symmetry, the input isomorphism must preserve the balanced coloring of the nodes. Thus, if we first color the input sets with a balanced coloring and then look for the subset of input isomorphisms that are also color preserving, then we have captured the isomorphisms that characterize the equitable partition without using the full input tree that is needed for the symmetry fibration. These isomorphisms are *symmetry isomorphisms* and form groupoids, not groups. We discuss this case next.

7.4 Symmetry isomorphisms and symmetry groupoid

In the same way that only some permutations can be permutation symmetries (i.e. automorphisms) of a network, only some input isomorphisms are symmetry isomorphisms. The set of all input isomorphisms has a natural algebraic structure: it is a groupoid. To explain

what this means and why it happens, we first generalize the notion to color-preserving isomorphisms (which are more important for our purposes):

Definition 7.2 Symmetry-isomorphism for colorings. Consider a network equipped with a balanced coloring. Then a *symmetry isomorphism* is a color-preserving input isomorphism. That is, it preserves input sets and maps nodes of a given color to nodes of the same color. This set is the *coloring symmetry groupoid* \mathcal{B}_ϕ of the network, for the coloring ϕ .

The set \mathcal{B}_ϕ is a groupoid because not all input isomorphisms can be composed. Two such isomorphisms α, β can be composed to define $\alpha\beta$ only when the codomain of β is the same as the domain of α . However, when color-preserving isomorphisms can be composed, the result is also color-preserving. Moreover, composition is associative when defined, and there are identity maps and inverses.

For example, in the sample network Fig. 7.4a, with the given coloring, the input isomorphism $\tau_{2 \rightarrow 3}$ is a symmetry isomorphism:

$$\tau_{2 \rightarrow 3} = \begin{pmatrix} 2 & 3 & 4 \\ \downarrow & \downarrow & \downarrow \\ 3 & 2 & 5 \end{pmatrix}, \quad (7.4)$$

because it maps red nodes to red nodes and a blue node to a blue node. Among the 28 input isomorphisms of the network in Fig. 7.4a only 10 are symmetry-isomorphisms, and they are given by the following maps:

$$\begin{aligned} \tau_{1 \rightarrow 1} &= \begin{pmatrix} 1 & 2 & 3 \\ \downarrow & \downarrow & \downarrow \\ 1 & 2 & 3 \end{pmatrix}, & \tau_{2 \rightarrow 2} &= \begin{pmatrix} 2 & 3 & 4 \\ \downarrow & \downarrow & \downarrow \\ 2 & 3 & 4 \end{pmatrix}, \\ \tau_{3 \rightarrow 3} &= \begin{pmatrix} 3 & 2 & 5 \\ \downarrow & \downarrow & \downarrow \\ 3 & 3 & 5 \end{pmatrix}, & \tau_{4 \rightarrow 4} &= \begin{pmatrix} 4 & 6 \\ \downarrow & \downarrow \\ 4 & 6 \end{pmatrix}, \\ \tau'_{1 \rightarrow 1} &= \begin{pmatrix} 1 & 2 & 3 \\ \downarrow & \downarrow & \downarrow \\ 1 & 3 & 2 \end{pmatrix}, & \tau_{2 \rightarrow 3} &= \begin{pmatrix} 2 & 3 & 4 \\ \downarrow & \downarrow & \downarrow \\ 3 & 2 & 5 \end{pmatrix}, \\ \tau_{3 \rightarrow 2} &= \begin{pmatrix} 3 & 2 & 5 \\ \downarrow & \downarrow & \downarrow \\ 2 & 3 & 4 \end{pmatrix}, & \tau_{5 \rightarrow 5} &= \begin{pmatrix} 5 & 6 \\ \downarrow & \downarrow \\ 5 & 6 \end{pmatrix}, \\ \tau_{4 \rightarrow 5} &= \begin{pmatrix} 4 & 6 \\ \downarrow & \downarrow \\ 5 & 6 \end{pmatrix}, & \tau_{5 \rightarrow 4} &= \begin{pmatrix} 5 & 6 \\ \downarrow & \downarrow \\ 4 & 6 \end{pmatrix}, \\ \tau_{7 \rightarrow 7} &= \begin{pmatrix} 7 & 3 \\ \downarrow & \downarrow \\ 7 & 3 \end{pmatrix}, & \tau_{6 \rightarrow 6} &= \begin{pmatrix} 6 & : \\ \downarrow & \\ 6 & : \end{pmatrix}. \end{aligned} \quad (7.5)$$

Now we consider the full set of input isomorphisms:

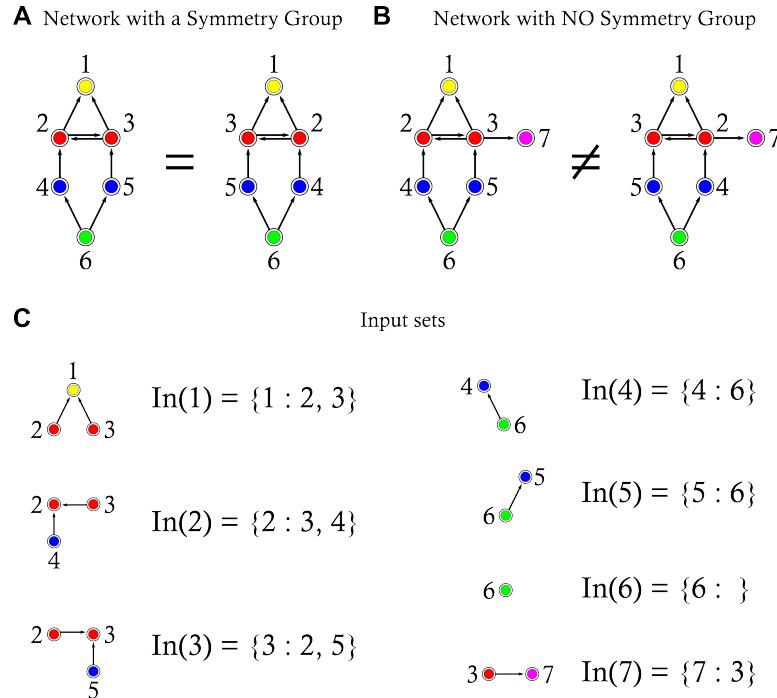


Fig. 7.4

From input sets to input isomorphisms and symmetry groupoids. (a) Example of a directed network. (b) Input sets of nodes of the network in (a), drawn with added source and target nodes for clarity. The input set $\text{In}(i)$ of node i is the set of all edges $j \rightarrow i$. For example, the input set of node 1 includes nodes 1, 2, and 3, i.e., $\text{In}(1) = \{1 : 2, 3\}$. (c) Input isomorphisms defining the groupoid of the network. (d) Input set symmetry isomorphisms defining the symmetry groupoid of the network.

Definition 7.3 Symmetry groupoid. The *symmetry groupoid* of the network is defined as the set of all symmetry isomorphisms of the network (Golubitsky and Stewart, 2006):

$$\begin{aligned} \text{Symmetry groupoid} = & \\ \{\tau_{i \rightarrow j} \mid \tau_{i \rightarrow j} \text{ is a symmetry isomorphism from } i \rightarrow j\}. & \end{aligned} \quad (7.6)$$

For example, the symmetry groupoid of the network in Fig. 7.4a is given by the isomorphisms of (7.5):

$$\begin{aligned} \text{Symmetry groupoid} = & \\ \{\tau_{1 \rightarrow 1}, \tau'_{1 \rightarrow 1}, \tau_{2 \rightarrow 2}, \tau_{2 \rightarrow 3}, \tau_{3 \rightarrow 3}, \tau_{3 \rightarrow 2}, \tau_{4 \rightarrow 4}, \tau_{4 \rightarrow 5}, \tau_{5 \rightarrow 5}, \tau_{5 \rightarrow 4}, \tau_{6 \rightarrow 6}, \tau_{7 \rightarrow 7}\}. & \end{aligned} \quad (7.7)$$

We think of symmetry-isomorphisms as the fibration analogs of group automorphisms. Then we can summarize the analogies between groups and groupoids via the following

correspondences:

$$\begin{aligned}
 \text{Groups} &\longrightarrow \text{Groupoids} \\
 \text{Permutations} &\longrightarrow \text{Input isomorphisms} \\
 \text{Automorphisms} &\longrightarrow \text{Symmetry isomorphisms}
 \end{aligned} \tag{7.8}$$

Symmetry-isomorphisms formalize the concept that nodes of the same color have input sets of the same color (Golubitsky and Stewart, 2006). Thus, in this definition of equivalence, nodes in the input sets lose their identity since which node sends the information is no longer important. What matters is that the information received by symmetrically related nodes is the same. In a balanced coloring, if two nodes i and j have the same color then their input sets $\text{In}(i)$ and $\text{In}(j)$ are isomorphic by a bijection that preserves colors of source nodes (and edge-types when edges can have more than one type).

In practice, the notion of a balanced coloring is primary, and its interpretation in groupoid language is included here to show that the groupoid formalism is equivalent to both the fibration viewpoint and that of balanced colorings. This also reinforces the analogy between the global symmetries captured by groups, and local symmetries captured by groupoids. Moreover, it establishes a ‘philosophical’ distinction between the group symmetries that are central to physics and the more general groupoid symmetries that we argue are natural for biology.

For this reason the mathematical formalism of groupoids, while theoretically important, has no practical use when finding balanced colorings and equitable partitions. In contrast, the fibration formalism, based on the full input tree, does not require a previous coloring to find the fibers, and this is the formalism that we will use in the remainder of this book in practical applications. The input tree determines only the minimal balanced coloring, but this is the most important coloring when determining building blocks of biological circuits.

An example of a balanced coloring is given for the network shown in Fig. 7.4d, and we have already checked that nodes of the same color receive inputs from nodes of the same colors.

Some groupoid symmetries are of a simple type, in which the input sets of the nodes concerned are exactly the same. Thus, they depend on the inbound neighboring list of nodes. However, more elaborated groupoid symmetries can occur for nodes that do not have exactly the same input set, but instead have equivalent input sets. The identification of these groupoid symmetries requires the input tree formalism of fibrations and iterative algorithms discussed in Chapter 13.

7.5 Example of groupoids and cluster synchronization

Fibrations can be viewed as a generalization of the classical notion of group symmetry. To compare the two, we recall Fig. 5.1 showing two simple networks, which in particular occur as synthetic genetic oscillators (Purcell et al., 2010). The metabolator network has a

global symmetry, which swaps the two nodes; this gives rise to a balanced coloring with an associated fibration. The Smolen oscillator has no group symmetry, but it does have a fibration. We show below that the synchronous dynamics of these two networks obeys the same equations, but the synchrony-breaking dynamics is typically very different in the two cases. This further exemplifies the difference between a symmetry group and fibration.

First, we write down the admissible ODEs for Fig. 5.1 (left). We list variables in the order: node, source of repressor connection, source of activator connection.

The admissible ODEs then have the form:

$$\begin{aligned}\dot{x}_1 &= f(x_1, x_1, x_2) \\ \dot{x}_2 &= f(x_2, x_2, x_1).\end{aligned}\quad (7.9)$$

Nodes 1 and 2 are synchronous if $x_1(t) = x_2(t)$ for all times t . Setting both equal to $y(t)$ the two equations become:

$$\begin{aligned}\dot{y} &= f(y, y, y) \\ \dot{y} &= f(y, y, y).\end{aligned}\quad (7.10)$$

Any solution of the common equation $\dot{y} = f(y, y, y)$ corresponds to a synchronous solution $(y(t))$ of (7.9), and *vice versa*.

For comparison, consider Fig. 5.1 (right). This is not symmetric. Listing variables in the same order, admissible ODEs have the form:

$$\begin{aligned}\dot{x}_1 &= f(x_1, x_2, x_1) \\ \dot{x}_2 &= f(x_2, x_2, x_1).\end{aligned}\quad (7.11)$$

Again, nodes 1 and 2 are synchronous if $x_1(t) = x_2(t)$ for all times t . Setting both equal to $y(t)$ the two equations once more reduce to (7.10), and any solution of the common equation $\dot{y} = f(y, y, y)$ corresponds to a synchronous solution $(y(t), y(t))$ of (7.9), and *vice versa*.

Figure 7.5 shows the input sets and input isomorphisms for these two networks. They appear very similar, but there are subtle differences. In particular, for the metabolator the map $\tau_{1 \rightarrow 2}$ swaps nodes 1 and 2 in all three columns, because this is a global symmetry of the network. In the Smolen oscillator, the map $\tau_{1 \rightarrow 2}$ swaps nodes 1 and 2 in the first column, but fixes both in the second.

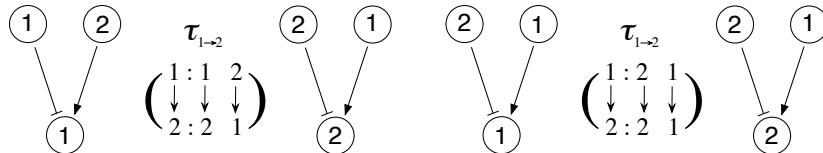


Fig. 7.5

Input sets and input isomorphisms. *Left:* Input sets for the metabolator network, and the unique input isomorphism between them. *Right:* Input sets for the Smolen oscillator network, and the unique input isomorphism between them.

We anticipate that the groupoid for both oscillators consists of an identity map for each input set, together with the corresponding map $\tau_{1 \rightarrow 2}$ and its inverse $\tau_{2 \rightarrow 1}$.

The existence of synchronous states for Fig. 5.1 (right) is governed by the analogous

fibration symmetry (Chapter 6), balanced coloring (Chapter 2 and this Chapter), or *input symmetry* described below in this Chapter. The crucial property here is that when certain nodes are required to have identical dynamics (synchrony) the entire system of admissible ODEs leads to a *consistent* set of equations for the synchrony classes (clusters).

For either oscillator in Fig. 7.5, the base of the fibration is a network with a single node with two self-loops: one for each type of arrow. The fibration maps both nodes to the node of the base, and maps arrows to preserve their type.

If the functions f in (7.9) and (7.11) are the same, then the ODE (7.10) for synchronous states is the same for both networks. Since symmetry groups and fibrations capture the same synchronized state in the metabolator and the Smolen oscillator, the question arises: What is the main difference captured by the fibration that is not captured by the symmetry group? The difference shows up when synchrony is broken, as we show for stability of equilibria in Section 8.8.

7.6 Algebraic formulation: lack of composition in groupoids

The composition law (totality or closure axiom in the axiomatic definition of abstract algebraic structures) is a fundamental axiom of groups. It states that if two transformations π and τ permute the nodes, then the composition $\pi \circ \tau$ is also some permutation of the nodes. This fundamental property of permutations, the closure axiom, is not always valid for the set of input isomorphisms defined in a given network.

Input isomorphisms in the groupoid formalism are the analogs of permutations in the theory of groups. However, whilst permutations can always be composed, input isomorphisms cannot, and for this reason they generate a groupoid rather than a group.

It is easy to verify that input isomorphisms cannot always be composed. For instance, $\tau_{2 \rightarrow 3}$ and $\tau_{3 \rightarrow 1}$, depicted in Fig. 7.6a, are input isomorphisms of the network concerned. The figure shows that the composition $\tau_{2 \rightarrow 3} \circ \tau_{3 \rightarrow 1}$ is well-defined, so this is an example of a legal composition. However, if we consider the same $\tau_{2 \rightarrow 3}$ but we try to compose it with the inverse input isomorphism $\tau_{1 \rightarrow 3}$ (Fig. 7.6b), we immediately see that this composition is not possible, since the codomain of $\tau_{2 \rightarrow 3}$ is node 3 and its input set and the domain of $\tau_{1 \rightarrow 3}$ is node 1. Thus, $\tau_{2 \rightarrow 3}$ and $\tau_{1 \rightarrow 3}$ cannot be composed.

In general, for two input isomorphisms to be composed, the codomain of the first and the domain of the second input isomorphisms, respectively, need to be the same. That is, two isomorphisms cannot be applied in succession $\sigma \circ \tau$ if the codomain of τ does not coincide with the domain of σ .

In a group of permutations of a set X , all domains and codomains are the same, namely X . This is why any pair of permutations can be composed. The groupoid structure arises when domains and codomains can differ, and this ultimately relates to the presence of special ‘base points’ for the input sets, namely the nodes that constitute the common targets, which lead to different domains and codomains.

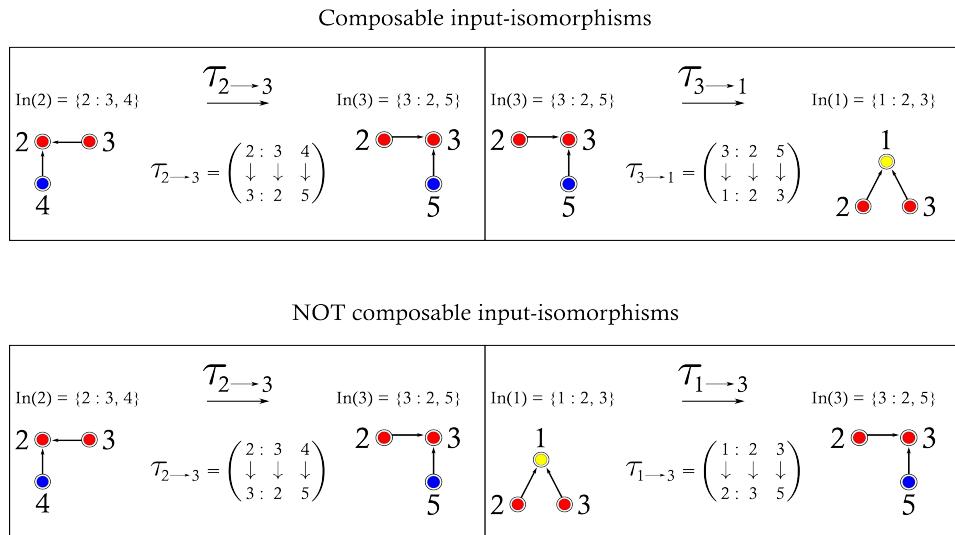


Fig. 7.6

Lack of composition law for input isomorphisms. *Top:* Example of composition of two input isomorphisms $\tau_{2 \rightarrow 3}$ and $\tau_{3 \rightarrow 1}$ of the network shown in Fig. 7.4b. Composition is possible if we apply first $\tau_{2 \rightarrow 3}$ and then $\tau_{3 \rightarrow 1}$, but not in the opposite order. More generally, for composition to be possible, the codomain of the first isomorphism must coincide with the domain of the second isomorphism. This condition is satisfied only if $\tau_{2 \rightarrow 3}$ is applied before $\tau_{3 \rightarrow 1}$, but not if they are applied in the reverse order. *Bottom:* Example of non composable input isomorphisms. In this case the isomorphisms $\tau_{2 \rightarrow 3}$ and $\tau_{1 \rightarrow 3}$ are never composable, since there is no match between the codomain of one and the domain of the other, no matter in which order they are applied.

Because the composition law need not hold, the set of input isomorphisms is not a group. Instead, these maps are groupoid transformations. However, other properties of a group persist. Groupoid transformations still have a well-defined inverse, an identity (indeed, often more than one identity), and composition satisfies the associative law *when composition is possible*. In summary, there are two groupoids associated with any network that is equipped with a coloring. One groupoid, say \mathcal{G} , consists of all input isomorphisms for all pairs of nodes. The other, say \mathcal{H} , consists of all input isomorphisms that map every node to a node of the same color. Here \mathcal{G} depends only on the graph, whereas \mathcal{H} depends on the coloring concerned. These coloring groupoids are all subgroupoids of the main groupoid.

The coloring symmetry groupoid, for a given coloring, lists the elements of \mathcal{H} . The groupoid \mathcal{G} contains extra maps. For example, in Fig. 7.4, nodes 1 and 2 are input isomorphic, because they each have two input arrows. So there is an input isomorphism $\tau_{1 \rightarrow 2}$. In fact, there are two of these, since we can swap the two arrows. Node 3 is input isomorphic to both 1 and 2, giving further input isomorphisms. So \mathcal{G} is larger than \mathcal{H} , or equivalently, \mathcal{H} is a subset of \mathcal{G} . However, in addition, \mathcal{H} is a groupoid in its own right, because com-

posing two maps the preserve colors also preserves colors. Technically speaking, \mathcal{H} is a *subgroupoid* of \mathcal{G} . Every coloring of the network gives rise to a subgroupoid in this manner.

Conversely, we can recover the coloring from the subgroupoid \mathcal{H} : two nodes i, j have the same color if some element of \mathcal{H} maps the input set of i to that of j . This construction defines a coloring of the nodes for any subgroupoid \mathcal{H} , but in general the resulting coloring need not be balanced. In order for the coloring thus defined to be balanced, the subgroupoid \mathcal{H} must satisfy extra conditions. These are somewhat technical, and will not be stated here, see (Golubitsky and Stewart, 2006); they restate in groupoid terminology the condition that the coloring reconstructed from \mathcal{H} must be balanced.

7.7 Relations between properties of fibrations, groups, and groupoids

We have associated a variety of mathematical structures with a graph, all related to concepts of symmetry; notably their symmetry fibrations, symmetry groups, and symmetry groupoids. There are numerous analogies between these fibrations, groups, and groupoids. Although the technical definitions of these terms differ considerably, these analogies let us ‘translate’ concepts from one structure to another. Table 7.2 presents an informal ‘dictionary’ showing how some of the basic concepts relate to each other. As with languages, subtle details can become lost in translation: that is, we should not expect every property of one concept to transfer unchanged to an analogous one. But bearing this dictionary in mind helps to organize important features and provides some motivation for them.

Property	Fibration	Group	Groupoid
preserved equivalence relation	input tree	adjacency	colored input set
transformation	input tree isomorphism	permutation	input set isomorphism
symmetry transformation	symmetry input tree isomorphism	automorphism	symmetry isomorphism
symmetry projection	symmetry fibration	quotient map	symmetry groupoid
cluster synchronization	fibers	orbits	balanced coloring
axiom of closure	not needed	needed	not needed
symmetry type	local	global	local
application	biology/AI	physics	biology

Table 7.2: **Informal Dictionary.** Properties of symmetry group, symmetry groupoid and symmetry fibration of a graph, indicating analogies between them and areas of science to which they usually apply.

7.8 Formal definition of a groupoid

Until now we have discussed groupoid notions informally, without defining their structure precisely. For completeness we now give a formal definition Brown (2006); Higgins (1971); Ibort and Rodriguez (2020), and formalize the relation between fibrations and groupoids.

This section can be skipped by anyone who does not want to see a formal treatment.

Definition 7.4 A *groupoid* A consists of:

- (a) A set O of *objects* i, j, k, \dots
- (b) For each $(i, j) \in O \times O$, a set $\Theta(i, j)$ of *morphisms*. This set may be empty.

Instead of writing $\theta \in \Theta(i, j)$, we often write $i \xrightarrow{\theta} j$, which is more intuitive.

The $\Theta(i, j)$ are assumed to be disjoint, or if necessary are made to be disjoint by redefining them. Thus the set of all morphisms is $\Theta = \bigsqcup \Theta(i, j)$ where the ‘squarecup’ indicates disjoint union.

(c) For each $i \in O$, the set $\Theta(i, i)$ contains a distinguished morphism ε_i . In particular, $\Theta(i, i) \neq \emptyset$ for all $i \in I$.

(d) There is a composition rule $(\theta, \phi) \mapsto \phi\theta$ whenever $i \xrightarrow{\theta} j \xrightarrow{\phi} k$, and $i \xrightarrow{\phi\theta} k$.

The following axioms must be satisfied:

- (e) *Identities*: If $i \xrightarrow{\theta} j$ then $\theta\varepsilon_i = \theta = \varepsilon_j\theta$.
- (f) *Inverses*: If $i \xrightarrow{\theta} j$ there exists $j \xrightarrow{\phi} i$ such that $\phi\theta = \varepsilon_i$ and $\theta\phi = \varepsilon_j$. We write $\phi = \theta^{-1}$.
- (g) *Associativity*: If $i \xrightarrow{\theta} j \xrightarrow{\phi} k \xrightarrow{\psi} l$ then $(\psi\phi)\theta = \psi(\phi\theta)$.

Writing products of subsets X, Y of Θ as XY , it is straightforward to deduce that

$$\Theta(j, k)\Theta(i, j) = \Theta(i, k) \quad (7.12)$$

If $\theta \in \Theta(i, j)$ then i is the *source* of θ and j is the *target* of θ .

Unlike a group, a groupoid has a different identity element for each object. It is a group if and only if it has a single object, or equivalently a unique identity element.

Example 13 Let the set of objects be $O = \{1, 2\}$. Define the sets of morphisms by:

$$\Theta(1, 1) = \{\varepsilon_1\} \quad \Theta(2, 2) = \{\varepsilon_2\} \quad \Theta(1, 2) = \{\alpha\} \quad \Theta(2, 1) = \{\beta\}$$

with composition defined by Table 1.

Figure 7.7 is a schematic diagram, showing these four morphisms and their relation to the objects 1 and 2. Composition is defined only when the corresponding arrows connect head to tail, forming a directed path. The table is then easy to construct. The identities are $\varepsilon_1, \varepsilon_2$, and $\beta = \alpha^{-1}$.

Network groupoids

We now state the connection between networks and groupoid theory in more formal terms.

Table 1. Composition table of groupoid in Example 13. Entry in row a , column b is ab .

	ε_1	ε_2	α	β
ε_1	ε_1	—	—	β
ε_2	—	ε_2	α	—
α	α	—	—	ε_2
β	—	β	ε_1	—

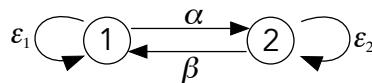


Fig. 7.7

Groupoid. Schematic diagram of a groupoid with objects 1, 2 and morphisms $\varepsilon_1, \varepsilon_2, \alpha, \beta$ composing only when the corresponding arrows connect head to tail.

Definition 7.5 Let G be a network, with C its set of nodes. The *network groupoid* \mathcal{B}_G is defined as follows:

The *objects* O of \mathcal{B}_G are the nodes $c \in C$, so $O = C$. (Alternatively the objects can be taken as the input sets $I(c)$ of these nodes, which are indexed by $c \in C$.)

The *morphisms* of \mathcal{B}_G in $\Theta(c, d)$ are the input isomorphisms $\beta : I(c) \rightarrow I(d)$. The usual notation in network dynamics is $B(c, d)$, but for this section we continue to use $\Theta(c, d)$.

The *network groupoid* \mathcal{B}_G of G is the disjoint union

$$\mathcal{B}_G = \bigsqcup_{c,d} \Theta(c, d)$$

with the operation of composition (where possible).

The *identity elements* of \mathcal{B}_G are the identity maps $\varepsilon_i : I(i) \rightarrow I(i)$.

The inverse α^{-1} of $\alpha : I(i) \rightarrow I(j)$ is its inverse as a mapping: $\alpha^{-1} : I(j) \rightarrow I(i)$.

It is easy to verify the groupoid axioms.

Coloring subgroupoids

A subgroupoid G' of a groupoid G is defined by taking a subset $O' \subseteq O$ and a subset of morphisms $\Theta'(i, j) \subseteq \Theta(i, j)$, with the condition that these subsets are chosen so that the groupoid axioms remain valid.

We can relate a fibration of a network G to a subgroupoid of its network groupoid \mathcal{B}_G by considering the corresponding balanced coloring (where colors \leftrightarrow fibers). Indeed, any coloring of G , balanced or not, is associated with a subgroupoid of \mathcal{B}_G :

Definition 7.6 Let G be a network with a coloring κ . An input isomorphism $\beta \in \Theta(c, d)$ is *color-preserving* if for all $e \in I(c)$,

$$\kappa(\mathbf{h}(\beta(e))) = \kappa(\mathbf{h}(e)) \quad \text{and} \quad \kappa(\mathbf{t}(\beta(e))) = \kappa(\mathbf{t}(e))$$

Write $\Theta^\kappa(c, d)$ for the set of all color-preserving elements of $\Theta(c, d)$.

A coloring is balanced if and only if

$$\kappa(c) = \kappa(d) \implies \Theta^\kappa(c, d) \neq \emptyset \quad (7.13)$$

Definition 7.7 Let G be a network with a coloring κ . The *coloring subgroupoid* \mathcal{B}_G^κ of κ is the set of all color-preserving input isomorphisms for κ . We have

$$\mathcal{B}_G^\kappa = \bigsqcup \Theta^\kappa(c, d)$$

The set \mathcal{B}_G^κ is a 'full subgroupoid' of \mathcal{B}_G ; that is, it contains all of the identity elements ε_i for $i \in C$.

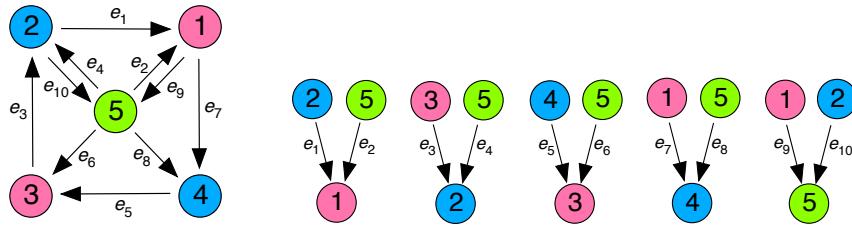


Fig. 7.8

Example of colored input sets. *Left:* Network with balanced coloring. *Right:* Input sets including head and tail nodes and their colors.

Example 14 Consider the 5-node network of Fig. 7.8 (left). The groupoid \mathcal{B}_G has 50 elements, the input isomorphisms β . We now consider which maps β preserve the colors of head and tail nodes of arrows, in the sense that $\kappa(\mathbf{h}(e)) = \kappa(\beta(\mathbf{h}(e)))$ and $\kappa(\mathbf{t}(e)) = \kappa(\beta(\mathbf{t}(e)))$. Clearly each identity element $\varepsilon_1, \dots, \varepsilon_5$ does so. There are precisely four others, in subsets $B(c, d)$ where $c \neq d$ and $\kappa(c) = \kappa(d)$:

$$\begin{pmatrix} e_1 & e_2 \\ \downarrow & \downarrow \\ e_5 & e_6 \end{pmatrix} \quad \begin{pmatrix} e_5 & e_6 \\ \downarrow & \downarrow \\ e_1 & e_2 \end{pmatrix} \quad \begin{pmatrix} e_3 & e_4 \\ \downarrow & \downarrow \\ e_7 & e_8 \end{pmatrix} \quad \begin{pmatrix} e_7 & e_8 \\ \downarrow & \downarrow \\ e_3 & e_4 \end{pmatrix}$$

Many results about balanced colorings can be interpreted using coloring subgroupoids, but the theory is technical. See (Stewart, 2024).

7.9 Hierarchy of algebraic structures

Algebraic structures like groups and groupoids can be defined in an axiomatic way. Four axioms defines the algebraic structures: closure (law of composition), invertibility, identity, and associativity, to which we add a fifth, commutativity, for completeness (Dixon and

Mortimer, 1996; Hamermesh, 2012). In the present study we consider the following algebraic structures which satisfy a number of axioms and are not required to satisfy others, see Table 7.3. The table shows a hierarchy of increasingly general algebraic structures, in which the axioms assumed become successively less restrictive.

Algebraic structure	Commutative	Closure	Inverses	Identity	Associative
Abelian group	✓	✓	✓	✓	✓
Non-Abelian group	✗	✓	✓	✓	✓
Groupoid	✗	✗	✓	✓	✓
Small Category	✗	✗	✗	✓	✓
Semigroupoid	✗	✗	✗	✗	✓

Table 7.3: Hierarchy of algebraic structures considered in the present study. There are other algebraic structures not listed here. ✓: required axiom. ✗: not required.

Starting from a commutative (Abelian) group which satisfies all five axioms, we drop one axiom at a time to create less restricted structures, resulting in the nested hierarchy:

Abelian groups → Non-abelian groups → Groupoids → Small Categories → Semigroupoids.

The most restrictive algebraic structure satisfies all the axioms listed: this is the Abelian Group. An example in physics is quantum electrodynamics (QED), with an Abelian gauge symmetry group $U(1)$. Abelian groups include graphs of atomic bonds and molecules.

A non-Abelian group is also called a non-commutative group. Examples in physics are the non-Abelian gauge theory of weak interactions, $SU(2)$, and the strong interaction, $SU(3)$. Physics uses other structures, such as Lie algebras and ‘quantum groups’—which despite the name are not groups—but there is a strong emphasis on groups and related structures. Permutations forming a symmetry group satisfy the axiom of closure, since they can be always composed and the result of this composition is again a permutation in the group.

Dropping the composition axiom, we obtain groupoid symmetries and fibrations, which describe many biological systems. The objects of the groupoid are (or correspond to) the nodes of the network. The morphisms are isomorphisms of input sets. These are ‘local symmetries’, indicating that certain *parts* of the network—the input sets of the nodes concerned—have the same structure. If there are input isomorphisms from node i to node j and from node j to node k :

$$i \xrightarrow{\alpha} j \quad j \xrightarrow{\beta} k$$

then these compose to give an input isomorphism from node i to node k :

$$i \xrightarrow{\beta\alpha} k$$

Otherwise, there is no natural way to compose input isomorphisms, which is why we obtain a groupoid, not a group.

The list of algebraic structures is completed with further structures obtained by removing one axiom at a time: a small category which has an identity and satisfies associativity, and finally a semigroupoid which satisfies only the axiom of associativity. We do not make use

of these last two, but groupoids are often viewed as categories in which every morphism has an inverse, and many groupoid structures generalize naturally to categories.

The transition from groups to groupoids is where life begins; biology is modeled by more general and more flexible structures than physics. We elaborate on this quasi-philosophical idea after Chapter 10, which reviews how groups describe physics with fiber bundles. These are similar to fibrations, but more restrictive.

Biological systems need more than the *existence* of cluster synchrony, because these systems must continue to function in a variable environment. The synchronous state must therefore be *stable*. In this chapter we discuss the stability of states of network ODEs, focusing on equilibria with cluster synchrony induced by fibrations. We begin with a quick review of asymptotic and linear stability. An equilibrium is asymptotically stable if nearby initial conditions converge to it under the flow of the ODE. It is linearly stable if all eigenvalues of the derivative (linearization, Jacobian matrix) have negative real part. Cluster synchrony has an important feature: the Jacobian decomposes into a block triangular matrix, whose eigenvalues are those of the two diagonal blocks. Thus the eigenvalues split into two subsets: synchronous eigenvalues, which correspond to synchrony-preserving perturbations, and transverse eigenvalues, which correspond to synchrony-breaking perturbations. We illustrate these concepts for a variety of genetic circuits using several types of models. Finally, we discuss the ‘master stability function’ of Pecora and Carroll (1998), initially developed to study stability of complete synchronization in Laplacian systems, and its generalizations to the problem of cluster synchronization in networks.

8.1 Biology needs stability

So far, we have concentrated on finding conditions for the *existence* of cluster synchrony. These conditions can be stated in four equivalent ways: a fibration symmetry, a balanced coloring, an equitable partition of nodes, the existence of a flow-invariant subspace for *any* admissible ODE. (A fifth viewpoint, groupoid symmetry, is not needed here.) For example, Fig. 2.1 shows a 3-node subnetwork of the GRN of *E. coli*. The fibration concerned is the map from this network to the 2-node network on the right hand side of the figure. The balanced coloring is given by the blue and pink colors. The equitable partition has two parts: {1} and {2, 3}. The synchrony subspace is

$$\Delta = \{(u, v, v) : u, v \in P^3\}$$

where each node has state space P .

All four formulations convey the same information with different emphasis. The key property for cluster synchrony is flow-invariance of Δ . This means that, dynamically, if we set the initial condition of the system to respect the fibers—that is, if the nodes in each fiber initially have the same state—then the same clusters remain synchronous for all forward time. However, this property alone is irrelevant to biology, because biological

systems must constantly adapt to a changing environment. What matters for biology is *stable* cluster synchronization; that is, *synchronizability*. Even when initial conditions are not synchronous, the dynamic trajectory should converge to the cluster state for all initial conditions in some neighborhood of that state.

That said, it is not possible to study the stability of a state unless that state exists. Moreover, the network topology does have one useful implication for stability analysis. The flow-invariance property associated with any cluster defined by a fibration splits the stability analysis into two parts: *synchrony-preserving* stability and *synchrony-breaking* stability. The first refers to stability to perturbations within the synchrony subspace; the second to stability transverse to the synchrony subspace. See Section 8.5.1.

The remarkable robustness of biological organisms and systems is a big question, and stability of an equilibrium (or a more complex dynamical state) is only part of the answer, since this refers to a specific model. A subtler issue is how the state of the system changes if the model itself is perturbed. This is ‘structural stability’, the main motivation for the topological approach to nonlinear dynamics arising from the work of Arnold (1989); Smale (1967); Thom (1975) from the 1960s.

For a given model, the concept of ‘stability’ has been given many different mathematical formulations, some of which we discuss in Section 8.2. They all have much the same intuitive implication: if the system is in an equilibrium state (in particular cluster synchrony) and it is then subjected to a small disturbance, it will settle back down towards either the original state or one that is very close to it—and in the network case, a state that has the same clusters, subject to the usual approximations in modeling assumptions. The main stability notions are *local*: they refer to sufficiently small perturbations of initial conditions; that is, to neighborhoods of the equilibrium. A more global indication of stability is the largest such neighborhood, the *basin of attraction*. For an equilibrium, or a more complex state—an ‘attractor’—the basin consist of all points whose dynamic trajectories limit on that attractor in forward time. The larger the basin, the ‘more stable’ the state becomes. Thus, for cluster synchrony, the size of the basin is one way to quantify synchronizability.

The existence of any given pattern of cluster synchrony depends only on the topology of the network; it is independent of the (admissible) dynamics because fibrations are purely graph-theoretic. However, the dynamics of the clusters depends on the choice of admissible ODE. The same goes for stability, as illustrated below. Stability is therefore a more complex issue than fibrations. There are two natural questions:

Q1: Is a given cluster synchronous state stable or unstable?

Q2: If it is stable, how big is its basin of attraction?

These questions need to be investigated more deeply to create a solid foundation for developing research applied to real networks.

Much research on Q1 has focused on complete synchrony, in which all nodes are synchronous: see Sections 8.1.1 and 8.10. References (Sorrentino et al., 2016; Siddique et al., 2018; Panahi et al., 2021; Lodi et al., 2021) address these questions for nontrivial clusters arising from fibrations, although the problem is better studied for automorphisms (Golubitsky et al., 1988; Golubitsky and Stewart, 2023; Pecora et al., 2014) where methods from group representation theory apply. We mention some articles that apply to fibrations in Section 8.1.1. Q1 is local: it concerns only an arbitrarily small neighborhood of the state

concerned. Q2 is harder because the basin of attraction is a global object: in particular, it can be the entire state space, in which case the cluster state is globally synchronizable.

However, existing results already have major consequences for information-processing networks, and show that synchronizability is possible under certain conditions. The existence of cluster synchrony gives a better understanding of the function of some information-processing networks. Synchronization reveals functional building blocks in biological networks Morone et al. (2020), which in turn shows that some circuits in genetic networks perform core logic computations Leifer et al. (2020).

Outline of Chapter

We first explain general concepts of stability with a specific focus on cluster synchrony of equilibria and illustrate them with examples. We show how flow-invariance of a synchrony space decomposes the eigenvalues of the Jacobian, which determine linear stability, into synchrony-preserving and synchrony-breaking parts.

We then apply these ideas to analyze particular examples of stability in some small genetic circuits, including bistability of the toggle-switch. This circuit has, for suitable parameters, two coexisting stable equilibria, and it can be made switch between them. In this respect it behaves just like an electronic flip-flop circuit.

We provide a short discussion of basic bifurcation theory, illustrated for the pattern of $\frac{1}{3}$ -period phase shifts that occurs in oscillatory states of the symmetric repressilator, a synthetic genetic circuit. Finally, we discuss the ‘master stability function’ of Pecora and Carroll (1998) to study stability of the complete synchronization for networks of coupled dynamical systems and its generalization to cluster synchronization developed in (Pecora et al., 2014; Sorrentino et al., 2016; Siddique et al., 2018; Panahi et al., 2021; Lodi et al., 2021).

8.1.1 Existing work on stability of cluster states

Some general theories have been developed with regard to *complete* synchronization under Laplacian dynamics. We briefly discuss two of them.

Stability of synchronous states has been widely studied for special models, such as the Kuramoto model (Kuramoto, 1984; Menara et al., 2019). Here the nodes are phase oscillators: the state space of a node is a topological circle, whose points define the phase. Nodes are considered to be synchronous if their states are approximately the same, and asynchronous if their states differ significantly. The ‘Kuramoto order parameter’ has been proposed to quantify synchronization (Nykamp, 2024). We do not discuss phase oscillators in this book, but they have many applications in both physical and biological science.

The second approach is the ‘master stability function’ of (Pecora and Carroll, 1998), which in its original formulation applies to complete synchronization and relies on a specific form of the admissible ODE, namely Laplacian dynamics. It has since been generalized to cluster synchrony; see Sections 8.10. and 8.11.

Complete synchrony is rare for most biological problems, and when it occurs it is often deleterious for an organism. For example, complete synchrony has been associated

with epileptic seizures. In contrast, cluster synchronization is common in biology—and important. In this chapter we deal with cluster synchronization by basing our analysis on general stability notions from nonlinear dynamics, which we explain through simple biological examples. In general, the stability issue is very complex and is highly sensitive to the choice of model and its parameters. Even regular 3-node networks illustrate these complexities (Leite and Golubitsky, 2006).

Boldi and Vigna (1997) study a network of processors that execute the same algorithm, showing the existence of self-stabilizing (decaying towards a synchronous state) algorithms and how they can be constructed in some cases. It is also known that for any given network, any cluster corresponding to a fibration/balanced coloring can occur as a stable equilibrium for *some* admissible ODE. This does not guarantee stability for biologically plausible model ODEs, but it does show that a full classification of fibrations is useful. See (Golubitsky and Stewart, 2023, Theorem 14.20).

Belykh and Hasler (2011, 2015) study synchronization in neural networks using algorithms that produce groupoids (which are equivalent to fibrations in networks, see Chapters 2 and 7). They find synchronous solutions and investigate the stability of these solutions. In their models, synchronous solutions are locally stable if the coupling term is big enough. Leifer et al. (2020) observe that synchronous solutions for genetic networks simulated by first order ODEs and first order DDEs are stable and have a very wide basin of attraction. In fact, solutions synchronize for any initial conditions given sufficient time; that is, if $x(t)$ and $y(t)$ simulate synchronous genes, then $\lim_{t \rightarrow \infty} (x(t) - y(t)) = 0$.

The stability of cluster synchronous steady states to synchrony-breaking perturbations is discussed in (Golubitsky and Stewart, 2023, Section 14.9). Some general results for networks with a certain type of feed-forward structure are obtained in (Stewart and Wood, 2024) for both equilibria and periodic states, with applications to animal locomotion (Stewart and Wood, 2025a) and neuron models (Stewart and Wood, 2025b). Other approaches and related results can be found in (Boccaletti et al., 2001; Pecora and Carroll, 1990; Polyak and Kvinto, 2017).

The study of stability of the cluster synchronous solution, either periodic or chaotic, has been investigated in (Sorrentino et al., 2016; Siddique et al., 2018; Panahi et al., 2021; Lodi et al., 2021). An extension of these studies to oscillators of different types can be found in (Sorrentino and Ott, 2007; Blaha et al., 2019; Della Rossa et al., 2020). The case of cluster synchronization originating from the presence of different delays over the network edges is described in (Nathe et al., 2020). Finally, cluster synchronization in adaptive networks is investigated in (Lodi et al., 2024).

8.2 Notions of stability

Recall that ‘synchronizability’ means the existence of a *stable* cluster synchrony pattern. The ultimate goal of this chapter is to provide methods that help to understand synchronizability of cluster synchronous states.

However, this issue is best approached by through general features of the stability of

equilibria for any system of ODEs, because the stability analysis for clusters depends on these results.

In nonlinear dynamics, the term ‘stable’ has many technically different meanings (Bhati and Szegö, 1970). Intuitively, a solution of an ODE is stable if it persists after any sufficiently small perturbation to initial conditions. ‘Persists’ means that the solution converges back towards the previous state as time tends to infinity, or, less strongly, tends towards something close to the original solution. For simplicity we focus on the stability of equilibria, which is the main concern in many biological applications.

To set the analysis in context, we define three standard notions of stability for equilibria, namely linear, exponential, and asymptotic stability. The first two of these are equivalent; the third is weaker. Linear and asymptotic stability are illustrated in Fig. 8.1. There are other stability notions; a common one is Liapunov stability, but we do not use this notion in this book. For further information see (Murray et al., 1993, Chapter 4) and (LaSalle, 1964; LaSalle and Lefschetz, 1961; Liapunov, 1893).

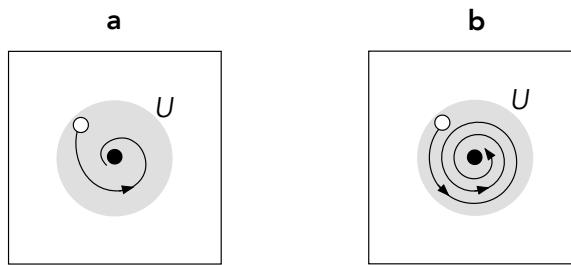


Fig. 8.1

Notions of stability for equilibria. (a) Exponential (= Linear) stability: There exists a ball U centered on the equilibrium, such that all trajectories starting inside U converge exponentially to the equilibrium. (b) Asymptotic stability: There exists a ball U centered on the equilibrium, such that all trajectories starting inside U converge to the equilibrium—possibly more slowly than exponential convergence.

To define these stability notions, consider the ODE $\dot{x} = f(x)$ for $x \in \mathbb{R}^n$, where $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$. Let x^* be an equilibrium point; that is, $f(x^*) = 0$. Recall that $\|x\|$ denotes the norm of a vector x in Euclidean space \mathbb{R}^n , defined as $\|x\| = \sqrt{x_1^2 + \dots + x_n^2}$ when $x = (x_1, \dots, x_n)$.

Linear Stability

The equilibrium x^* is *linearly stable* if all eigenvalues of the Jacobian Df evaluated at x^* have negative real part.

Exponential Stability

The equilibrium x^* is *exponentially stable* if the rate of convergence is exponential. That is, there is a neighborhood U of x^* and constants $K, \alpha > 0$ such that $\|x(t) - x^*\| < Ke^{-\alpha t}$ for all $x(0) \in U$.

Asymptotic Stability

The equilibrium x^* is *asymptotically stable* if all solutions with initial conditions sufficiently close to x^* converge to x^* in forward time. That is, there is a neighborhood U of x^* such that $\|x(t) - x^*\| \rightarrow 0$ as $t \rightarrow +\infty$ for all $x(0) \in U$. Unlike exponential stability, the rate of convergence to x^* can be very slow.

Exponential stability is equivalent to linear stability, and either of them implies asymptotic stability, but the converse is not valid in general. Linear stability is equivalent to exponential stability.

'Liapunov stability', is a weaker stability notion still, in which solutions that start sufficiently close to x^* remain close to it in forward time. We do not give the formal definition since we avoid this concept of stability in favor of stronger ones. This notion goes back to (Liapunov, 1893) and is the central topic of (LaSalle and Lefschetz, 1961). It becomes useful when there is a suitable 'Liapunov function', which guarantees Liapunov stability. Such functions can sometimes be constructed when other stability notions are difficult to establish. See (Hirsch and Smale, 1974; Stewart and Wood, 2024) for further information.

Stability theory for periodic orbits is analogous, and reduces to the theory for an equilibrium by taking a 'Poincaré section' transverse to the periodic orbit and considering the corresponding 'Poincaré map', which maps a point in the section to the first point at which its trajectory returns to the section. Linear stability then becomes what classically is known as 'Floquet theory'. Nonlinear flows can also produce chaotic oscillations. Chaotic trajectories show sensitive dependence on the initial condition, i.e., a very small change in the initial condition results in a completely different trajectory for large enough time. Despite this characterization, chaotic oscillators can synchronize and linear stability can be used to assess whether synchronous chaotic oscillations are stable or not (Pecora and Carroll, 1998). This topic is discussed at the end of this Chapter.

8.3 Stability of equilibria of linear ODEs

In what follows we consider only linear stability of an equilibrium. This is the strongest notion of stability, and it implies exponential, asymptotic, and Liapunov stability. An advantage of linear stability is that it is determined by the eigenvalues of the Jacobian (derivative, Fréchet derivative) at the equilibrium point, and so reduces to linear algebra. Excellent algorithms exist for numerical computations.

We begin with the simplest case: equilibria of linear ODEs. This case provides motivation for the general theory and leads naturally to stability conditions for more complicated systems. Consider the following example, in which the equations 'decouple':

$$\dot{x} = ax \quad \dot{y} = by$$

where $a, b \neq 0$ are constants. There is an explicit solution:

$$x(t) = e^{at} x_0 \quad y(t) = e^{bt} y_0$$

where x_0, y_0 are initial conditions at $t = 0$. That is, $x(0) = x_0, y(0) = y_0$.

The unique equilibrium is the origin $(0, 0)$. If $a, b < 0$, all solutions tend towards the origin as t tends to infinity. If either of $a, b > 0$, most solutions tend to infinity. So the condition for $(0, 0)$ to be a stable equilibrium is that $a, b < 0$. We call the origin a *sink* in this case. Otherwise the origin is a *saddle* or a *source*, as in Fig. 8.2.

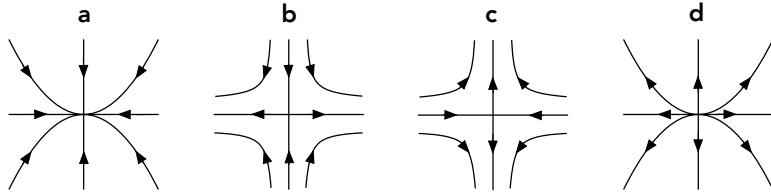


Fig. 8.2

Phase portraits for a decoupled linear ODE in \mathbb{R}^2 . (a) $a < 0, b < 0$ (sink). (b) $a > 0, b < 0$ (saddle). (c) $a < 0, b > 0$ (saddle). (d) $a > 0, b > 0$ (source).

We recast this analysis in matrix form. Let

$$X = \begin{bmatrix} x \\ y \end{bmatrix} \quad A = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}.$$

Then the equation becomes

$$\dot{X} = AX$$

so that

$$X = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} e^{at} & 0 \\ 0 & e^{bt} \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = e^{At} X_0.$$

We recognize the constants a, b as the eigenvalues of A . More generally, if we replace A by any diagonalizable matrix, a linear change of coordinates brings it to the form just analyzed, and again stability depends on the signs of the eigenvalues. More precisely, it depends on the signs of their real parts, because eigenvalues can occur in complex pairs $c \pm id$, and $e^{c \pm id} = e^c e^{\pm id}$. The term $e^{\pm id}$ corresponds to an oscillation of constant amplitude, and the factor e^c damps the amplitude down to zero when $c < 0$, but blows it up to infinity if $c > 0$. The characterization in terms of the signs of the real parts of the eigenvalues applies even when the matrix is not diagonalizable, using Jordan normal form.

A general linear ODE on \mathbb{R}^n has the form

$$\dot{X} = AX$$

where now $X = (x_1, \dots, x_n) \in \mathbb{R}^n$ is thought of as a column matrix (equal to $[x_1, \dots, x_n]^T$ where T indicates the transpose) and $A = [a_{ij}]$ is an $n \times n$ matrix. The general solution is

$$X(t) = e^{At} X_0. \tag{8.1}$$

Considering the eigenvalues leads to:

Theorem 8.1 *The origin is a stable equilibrium of (8.1) if all eigenvalues of A have negative real part. It is unstable if any eigenvalue of A has positive real part.*

If any eigenvalue of A has zero real part, the stability is not determined. This case is ‘non-generic’ in an individual ODE; that is, it typically does not occur. In Section 8.9 we see that in a nonlinear ODE, this case indicates the possibility of a ‘bifurcation’, and higher-order terms are needed to determine stability at the bifurcation point.

8.4 Stability of equilibria of nonlinear ODEs

In general, linear models are too simple and artificial to capture genuine biological behavior, so most biological models are nonlinear. However, linear stability theory for nonlinear ODEs is—as the name suggests—modeled on the linear case. Let $X = (x_1, \dots, x_n) \in \mathbb{R}^n$, and consider the ODE

$$\dot{X} = F(X) \quad (8.2)$$

where $F : \mathbb{R}^n \rightarrow \mathbb{R}^n$. In coordinates, this takes the form

$$\begin{aligned}\dot{x}_1 &= f_1(x_1, \dots, x_n) \\ \dot{x}_2 &= f_2(x_1, \dots, x_n) \\ &\vdots \\ \dot{x}_n &= f_n(x_1, \dots, x_n)\end{aligned}$$

where $F = (f_1, \dots, f_n)$.

Assume that (8.2) has an equilibrium $X_0 \in \mathbb{R}^n$, so that $F(X_0) = 0$. (Here we use 0 for the zero element of \mathbb{R}^n , which in coordinates is $(0, 0, \dots, 0)$.) Form the Taylor series of F to order 1:

$$F(X_0 + X) = F(X_0) + JX + O(2) \quad (8.3)$$

where $O(2)$ indicates an error term of order $\|X\|^2$, and J is the *Jacobian matrix* (or (Fréchet derivative) evaluated at X_0):

$$J = DF|_{X_0}.$$

In components, J is the matrix of partial derivatives of components f_i of F :

$$J = (Df|_{X_0})_{ij} = \frac{\partial f_i}{\partial x_j}(X_0).$$

Intuitively, the $O(2)$ term in (8.3) becomes arbitrarily small compared to JX as $\|X\| \rightarrow 0$, so we expect solution trajectories $X(t)$ near X_0 to resemble those of the *linearization*

$$\dot{X} = JX.$$

This can be proved to be the case provided the equilibrium is *hyperbolic*: J has no eigenvalues on the imaginary axis. The Kupka–Smale Theorem states that hyperbolicity is a ‘generic’ (typical) property in a general dynamical system (Kupka, 1963, 1964; Smale, 1963; Peixoto, 1966).

Bearing Theorem 8.1 in mind, we state:

Definition 8.2 *The equilibrium X_0 of (8.2) is linearly stable if all eigenvalues of the Jacobian matrix, evaluated at X_0 , have negative real part.*

Linear stability implies asymptotic stability for nonlinear ODEs. We therefore take linear stability as the most convenient stability notion for equilibria—in particular, for cluster synchronous equilibria.

8.5 Two examples

We now illustrate the use of the eigenvalues of the Jacobian to establish stability (or not) of equilibria for two simple networks: the Goodwin circuit and a circuit with two different fibrations. First, we set up suitable notation for the second of these examples, Example 16. We use the same notation later in this chapter.

Notation for partial derivatives

In the multigraph setting it is sometimes necessary to use notation for partial derivatives that differs from the familiar $\partial f / \partial x_j$ notation. Let $f : \mathbb{R}^m \rightarrow \mathbb{R}$ be one component of an ODE on \mathbb{R}^m . We denote the partial derivative of f with respect to the i th variable by f_i . (An alternative, often used, is $\partial_i f$.)

We explain why we use this notation instead of the more usual $\partial f / \partial x_j$. When different input arrows have different tail nodes, f_i is equal to $\partial f / \partial x_i|_{(x_0, y_0)}$, as usual. However, this notation is ambiguous when the same tail node occurs for more than one arrow. For example, suppose that one component of the admissible ODE is $\dot{x} = f(x, y, y)$. Then $\partial f / \partial y$ could refer either to the second or third variable. Technically we should define f using dummy variables; for example let $f(u, v, w) = u^2 + v + w$. Then the three partial derivatives $\partial f / \partial u$, $\partial f / \partial v$, and $\partial f / \partial w$ are uniquely defined: they are $2x, 1, 1$ respectively. When we substitute $u = x$ and $v = w = y$ we get $f(x, y, y) = x^2 + 2y$. Now

$$\frac{\partial f(x, y, y)}{\partial y} = \frac{\partial(x^2 + 2y)}{\partial y} = 2$$

is uniquely defined, but $\partial f / \partial y$ is not, since f depends on u, v, w , not on y . The problem is that f is doing double duty: as the name of the function and as the result of making the substitution.

Indeed, if all we know is that, say, $f(x, y, y) = x^2 + 2y$, then f is not unique. Perhaps $f(u, v, w) = u^2 + 2v$, or $f(u, v, w) = u^2 + 3v - w$.

Using subscripts to denote partial derivatives of $f(u, v, w)$ avoids this issue by specifying which variable of f is concerned. In detail,

$$\frac{\partial f(x, y, y)}{\partial y} = f_2(x, y, y) + f_3(x, y, y)$$

by the chain rule for partial derivatives. In the above example $f_2 = f_3 = 1$, so this formula gives $1 + 1$, which equals 2, as before.

This notation should not be confused with f_i for the i th component of a map $F = (f_1, \dots, f_i)$. In that case the partial derivatives would be denoted $f_{i,j}$, or for greater clarity, $\partial_j f_i$.

Example 15 Perhaps the simplest example is the Goodwin circuit, often called the Goodwin oscillator because suitable models generate oscillatory behavior (Goodwin, 1963). This ‘circuit’ comprises a single node that autoregulates via a repressor arrow. It is also one of the network motifs—called negative auto-regulation loop—identified in bacterial genetic networks in (Alon, 2019). Suitable models generate oscillatory behavior, hence the name, but here we consider equilibria.

We assume a model in which the state of each node is determined by two variables: the mRNA concentration x^R and the protein concentration x^P . The most general admissible ODE is then:

$$\dot{x}^R = f(x^R, x^P) \quad \dot{x}^P = g(x^P, x^R).$$

Assume there is an equilibrium state $X_0 = (x_0^R, x_0^P)$. The Jacobian at X_0 is

$$J = \begin{bmatrix} \frac{\partial f}{\partial x^R} & \frac{\partial f}{\partial x^P} \\ \frac{\partial g}{\partial x^R} & \frac{\partial g}{\partial x^P} \end{bmatrix}$$

which must be evaluated at X_0 to specify the linear stability of X_0 .

For suitable f and g this can be any 2×2 matrix. So we cannot say much about it without specifying f and g . To make progress, consider the ‘special model’

$$\dot{x}^R = -\delta x^R + \frac{1}{1 + (x^P)^2} \quad \dot{x}^P = -\alpha x^P + \beta x^R.$$

Here the term $-\delta x^R$ represents mRNA degradation, $-\alpha x^P$ represents protein degradation, β represents the rate at which mRNA produces protein, and $\frac{1}{1 + (x^P)^2}$ is a ‘Hill function’, see Section 8.7.1. The constants α, β, δ are all positive for this circuit.

It is convenient to simplify notation by setting

$$x^R = x \quad x^P = y$$

so the equations become

$$\dot{x} = -\delta x + \frac{1}{1 + y^2} \quad \dot{y} = -\alpha y + \beta x.$$

Equilibria therefore satisfy

$$0 = -\delta x + \frac{1}{1 + y^2} \quad 0 = -\alpha y + \beta x.$$

The second equation gives $y = \frac{\beta}{\alpha}x$. Let $\gamma = \frac{\beta}{\alpha}$ and write this relation as $y = \gamma x$ to keep the formulas simple. Then the first equation becomes

$$\delta x = \frac{1}{1 + \gamma^2 x^2}$$

which simplifies to a cubic equation for x :

$$\gamma^2 \delta x^3 + \delta x = 1.$$

Since $\gamma, \delta > 0$ this equation has a unique real solution x_0 , and it is positive. Positivity is a biological requirement since x is the RNA concentration. The equilibrium is therefore at (x_0, y_0) where $y_0 = \gamma x_0$.

The Jacobian is

$$J = \begin{bmatrix} -\delta & \frac{-2y}{(1+y^2)^2} \\ \beta & -\alpha \end{bmatrix} \Big|_{(x_0, y_0)}.$$

A standard result about 2×2 matrices states that the eigenvalues have negative real parts if and only if the trace is negative and the determinant is positive. Thus stability requires

$$-\delta - \alpha < 0 \quad \alpha\delta + \frac{2y_0}{(1+y_0^2)^2} > 0.$$

Since $\alpha, \delta < 0$ the first inequality holds. Moreover, $y_0 = \gamma x_0$, so $y_0 > 0$; also $\alpha\delta > 0$. Therefore the second inequality holds. We conclude that there is always a unique equilibrium, which is positive, and it is linearly stable for all parameters $\alpha, \beta, \delta > 0$. In more complex models, this state can become unstable, creating oscillations.

Even for this simple model, we run into a cubic equation. For more complex networks and models, explicit solutions are very unlikely, so numerical methods must be used. However, sometimes partial information can be obtained analytically.

Example 16 Next we consider Fig. 8.3, an artificial network introduced to illustrate the mathematics for cluster synchrony. There are two possible divisions of this graph into fibers: one is minimal and one is not.

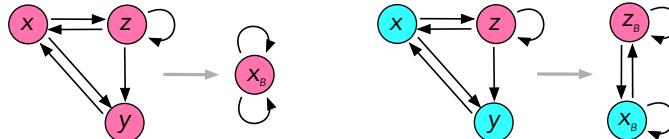


Fig. 8.3

Two fibrations of a 3-node network. *Left:* Minimal base: one color; base has one node. *Right:* Two colors; base has two nodes.

In the minimal case, all nodes are synchronous, $x(t) = y(t) = z(t) = x_B(t)$, on the left of the figure. On the right is a non-minimal cluster state, $x(t) = y(t) = x_B(t), z(t) = z_B(t)$. For illustrative purposes, suppose that these states are equilibria, and assume that the state of each node is determined by a single variable; that is, $x, y, z \in \mathbb{R}$. Admissible ODEs are:

$$\begin{aligned}\dot{x} &= f(x, \bar{y}, \bar{z}) \\ \dot{y} &= f(y, \bar{x}, \bar{z}) \\ \dot{z} &= f(z, \bar{x}, \bar{z})\end{aligned}$$

where the overline indicates that f is symmetric in its second and third variables.

The restriction of the ODE to the synchrony subspace is $\dot{u} = \overline{f}(u, u, u)$. A fully synchronous equilibrium $X_0 = (u, u, u)$ is a solution of $\overline{f}(u, u, u) = 0$. The Jacobian of the restricted ODE on the synchrony subspace $\Delta = (u, u, u) : u \in \mathbb{R}$ is

$$J|_{\Delta} = f_1 + f_2 + f_3$$

where, as discussed above, f_i is the partial derivative of f with respect to the i th variable. (We cannot use the notation $\partial f / \partial u$ since the same variable u occurs three times. See section 8.7.4.) Thus the completely synchronous state is linearly stable to *synchrony-preserving* perturbations if and only if $f_1(X_0) + f_2(X_0) + f_3(X_0) < 0$. The symmetry condition on f implies that $f_2(X_0) = f_3(X_0)$ because X_0 is fixed by the symmetry, so this becomes $f_1(X_0) + 2f_2(X_0) < 0$.

The Jacobian of the full ODE on \mathbb{R}^3 is

$$J = \begin{bmatrix} f_1 & f_2 & f_3 \\ f_2 & f_1 & f_3 \\ f_2 & 0 & f_1 + f_3 \end{bmatrix}.$$

Again $f_2(X_0) = f_3(X_0)$, so this simplifies to

$$J = \begin{bmatrix} f_1 & f_2 & f_2 \\ f_2 & f_1 & f_2 \\ f_2 & 0 & f_1 + f_2 \end{bmatrix}.$$

The eigenvalues are

$$f_1 + 2f_2, f_1, f_1 - f_2$$

all evaluated at X_0 .

It is easy to see that either of f_1 or $f_1 - f_2$ (or both) can be positive, while keeping $f_1 + 2f_2$ negative. Thus the fully synchronous state X_0 can be stable with respect to synchrony-preserving perturbations, yet unstable to synchrony-breaking perturbations.

On the other hand, all three eigenvalues can be negative, in which case X_0 is stable for the full ODE on \mathbb{R}^3 .

This calculation shows that:

- (a) The stability of an equilibrium depends on the details of the model.
- (b) Stability with respect to synchrony-preserving perturbations, although necessary for stability of the cluster state, need not be sufficient.

The cluster state with two colors can be analyzed in the same manner. Now $X_0 = (u, u, v)$ and $\Delta = \{(u, u, v) : u, v \in \mathbb{R}\}$, which is 2-dimensional. There are two equilibrium conditions for X_0 , namely $f(u, u, v) = 0$ and $f(v, u, v) = 0$. Since the second and third coordinates of X_0 are different, we no longer have $f_2(X_0) = f_3(X_0)$. The eigenvalues of the Jacobian J for the full ODE on \mathbb{R}^3 turn out to be

$$f_1 + f_2 + f_3 \quad f_1 \quad f_1 - f_3$$

and these must be evaluated at X_0 . We recognize the first two as the synchronous eigenvalues—those of the restricted Jacobian $J|_{\Delta}$. These eigenvalues determine stability with respect to synchrony-preserving perturbations. The third is the transverse eigenvalue, determining stability with respect to synchrony-breaking perturbations. Again the stability depends heavily on the model equations, and statements (a) and (b) above apply.

Both cluster states for this network can be stable for the same model and parameters. To see this, choose f so that $a, b, c < 0$, $a < b$ and $a < c$. It is also possible for either state to be stable while the other is unstable, or for both to be unstable.

8.5.1 Synchronous and transverse eigenvalues and eigenvectors

Example 16 illustrates a more general theorem, in which the stability of a cluster equilibrium is decomposed into two parts: stability with respect to synchrony-preserving perturbations, and stability with respect to synchrony-breaking perturbations. This decomposition arises from the flow-invariance of the synchrony subspace Δ . It simplifies the calculation of eigenvalues and eigenvectors by decomposing state space P into two parts: Δ itself, and the quotient space P/Δ . In general Δ need not have a flow-invariant complement, but the eigenvalues that do not correspond to eigenvectors lying in the synchrony subspace can be found from the vector space quotient.

Because Δ is flow-invariant, the Jacobian can be placed in block-triangular form by choosing a basis for Δ and then extending it to a basis for P . Then

$$J = \begin{bmatrix} A & 0 \\ B & C \end{bmatrix}$$

for suitable matrices A, B, C . The eigenvalues of J are therefore those of A and those of C , which are smaller matrices than J , aiding computations.

The matrix A is the restriction $J|_{\Delta}$ of J to the synchrony space. The matrix $C = \vec{J}$ is that for the action of J on the quotient space P/Δ . Stability with respect to synchrony-preserving perturbations is determined by the eigenvalues of $A = J|_{\Delta}$. The remaining eigenvalues of J are those of $C = \vec{J}$, and determine the stability with respect to synchrony-breaking perturbations.

Definition 8.3 *The transverse Jacobian at x_0 is the linear map $\vec{J}|_{x_0}$ induced by $J|_{x_0}$ on the vector space quotient P/Δ .*

The transverse eigenvalues of $J|_{x_0}$ are precisely those of $\vec{J}|_{x_0}$.

The synchronous eigenvalues of $J|_{x_0}$ are precisely those of $J|_{\Delta}$.

Theorem 8.4 *The equilibrium x_0 is linearly stable if and only if all synchronous eigenvalues have negative real parts and all transverse eigenvalues have negative real parts.*

For nontrivial clusters, both of the matrices $J|_{x_0}$ and $\vec{J}|_{x_0}$ are smaller than $J|_{x_0}$, so this decomposition simplifies the calculation of the eigenvalues, hence the linear stability of the equilibrium. A simple formula for $\vec{J}|_{x_0}$ is stated and proved in (Golubitsky and Stewart, 2023, Theorem 14.23), but we do not use this here.

8.6 Some small genetic circuits

In Part II we will identify building blocks of gene regulatory networks through the fibration symmetries of the underlying biological graph. In (Stewart et al., 2024) we analyze analytically the cluster synchronized solutions, and their stability, for seven of these ‘circuits’, shown in Fig. 8.4. In this chapter we summarize the results for five of these circuits, omitting the feed-forward fiber and Fibonacci fiber circuits. Their biological significance, and in particular their appearance in the *E. coli* GRN, is discussed in Chapter 15. Arese-Lucini et al. (2020) discusses stability in ecosystems.

All of these circuits occur as functional and synchronous building blocks in real or synthetic genetic networks. The lock-on, toggle-switch, Smolen, feed-forward fiber, and Fibonacci fiber circuits occur in living organisms; they were first observed in *E. coli* in (Morone et al., 2020) and later in other organisms from yeast to humans (Leifer et al., 2020). The sixth is the metabolator, a GRN incorporating metabolic effects (Fung et al., 2005), and the seventh is the synthetic GRN repressilator (Elowitz and Leibler, 2000).

In Section 8.8 we discuss the lock-on circuit as a typical example. In Section 8.8.3 we compare the Smolen circuit with the very similar metabolator circuit (Fung et al., 2005; Purcell et al., 2010). Section 8.8.2 discusses bistability of the toggle-switch. The repressilator’s oscillatory states are discussed in Section 8.9.2. The feed-forward fiber and Fibonacci fiber circuits are analyzed from the same viewpoint in (Stewart et al., 2024).

For each of these circuits, the network topology admits a fibration symmetry, leading to the existence of synchronous cluster states. These states can be steady, periodic, or chaotic, depending on the model. In Section 8.8 we show how the network framework permits a systematic analysis of certain aspects of dynamics and bifurcations in these GRN circuits. Similar methods apply to other small circuits. The main aim in this chapter is to illustrate several mathematical ideas for analyzing circuit dynamics on examples of

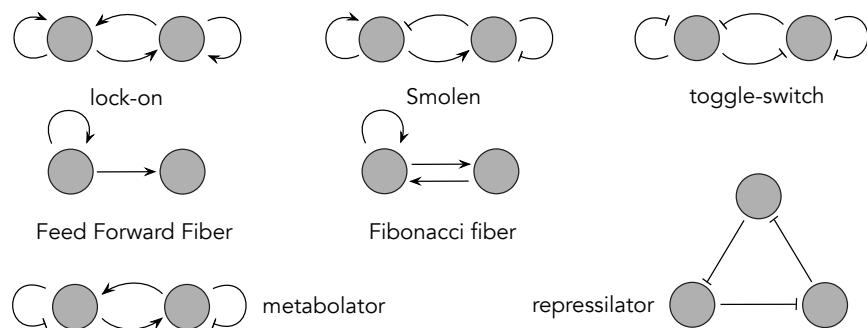


Fig. 8.4

Seven small circuits that occur as functional and synchronous building blocks in GRNs. The lock-on, Smolen, toggle-switch, feed-forward fiber, and Fibonacci fiber are bases for circuits found in the *E. coli* GRN. The repressilator is a synthetic GRN that sustains periodic oscillations. The metabolator is a GRN that includes metabolic effects.

biological relevance. We do not aim at a systematic analysis of all the dynamic behavior that can occur in these circuits, which in any case depends on the choice of model. We focus on general principles that give insight into the stability of equilibria, and emphasize the distinction between synchrony-preserving and synchrony-breaking stability. We show how fibration symmetry simplifies the calculation of the eigenvalues of the Jacobian at a cluster equilibrium.

In Section 8.9 we recall some basics of bifurcation theory. A bifurcation occurs when varying a parameter causes a stable state to become unstable, in which case the system typically moves to some other stable state. The two main types of local bifurcation from an equilibrium are steady-state bifurcation, resulting in new equilibria, and Hopf bifurcation, resulting in a periodic oscillation. Bifurcations can either preserve or break cluster synchrony, depending on the corresponding critical eigenvectors. We apply these ideas to some of the six circuits in this section, after we have set up the necessary mathematical background. Similar methods apply to any small circuit with fibration symmetries, but the calculations may be more complicated.

8.7 Protein/mRNA network structure

A gene regulatory network (GRN) is often referred to as a transcriptional regulatory network (TRN), although the former is more general than the latter. We use both terms here. The usual format for GRN diagrams in the biological literature uses one node per gene. However, for some mathematical purposes it can be more convenient to work with a modified representation of gene regulatory networks, so that standard models occur as admissible ODEs in a manner that makes the different roles of transcription and translation explicit.

To do this we use a protein-mRNA network (PRN) representation (Stewart et al., 2024), which splits each gene into two nodes: one for mRNA and one for protein. The PRN format effectively gives each gene node a 2-dimensional node space with a restriction on the node dynamic: the arrow from the mRNA component to the protein component must be inside that node. In contrast, arrows from the protein component can connect to mRNA nodes of other genes via regulation.

This unorthodox representation is, of course, implicit in the simpler networks commonly used in biology, but there are mathematical advantages in making it explicit. An arbitrary 2-dimensional node state space permits more general ODEs that fail to incorporate important biological information: namely, that translation of mRNA to protein occurs within a single gene, but transcription of DNA to mRNA, regulated by a protein, can link different genes. Subsection 8.7.2 draws useful conclusions from this structure.

8.7.1 Hill functions

We will illustrate some stability calculations with simple, explicit models based on Hill functions. A *Hill function* $H : \mathbb{R}^{\geq 0} \rightarrow \mathbb{R}$ is any function of the form

$$H^-(z) = \frac{1}{1+z^n} \quad 1 \leq n \leq \infty. \quad (8.4)$$

(Since z is a concentration we require $z \geq 0$.) This function has a ‘sigmoid’ shape and is monotonic decreasing for all n . We have $H^-(0) = 1$ and $H^-(z) \rightarrow 0$ as $z \rightarrow \infty$. Therefore this form represents *repression* of gene j on gene i when used in the context of $f_i^R(x_j^P)$. The minus sign in H^- is a reminder of this. For *activation* we use

$$H^+(z) = 1 - H^-(z) = \frac{z^n}{1+z^n}. \quad (8.5)$$

This function also has a sigmoid shape, but now it is monotonically increasing for all n . We have $H^+(0) = 0$ and $H^+(z) \rightarrow 1$ as $z \rightarrow \infty$. This form represents *activation* of gene j on gene i , with the plus sign as a reminder.

The function H^\pm can be modified in several ways: change the value of n ; scale the variable z considering $H^\pm(\mu z)$ for a parameter $\mu > 0$; scale the *value* of H^\pm by using vH^\pm ; move the origin by translating z to $z + z_0$ for some z_0 . Usually we take $n = 2$, both for simplicity and because it is relatively typical.

8.7.2 Bipartite structure

Definition 8.5 *A graph is bipartite if its nodes split into two components, so that the head and tail of any arrow lie in different components.*

Every PRN is bipartite. The two components are the mRNA nodes and the protein nodes. The mRNA nodes connect to proteins only via internal translation arrows, which we draw with wavy lines, while proteins connect to mRNA either within a single gene or between distinct genes via transcription arrows, which we draw with solid lines, either activator or repressor. (The PRN structure does not consider other types of interaction between genes, but modified network structures can represent these.)

We distinguish two types of model:

- *General model:* the most general ODEs that are compatible with the network structure; that is, the admissible ODEs for the PRN.
- *Special model:* this common type of model uses a Hill function and a linear degradation term for mRNA concentration, and a linear function and linear degradation term for protein concentration. Activators and repressors are represented by the signs of coefficients and the choice of Hill functions. Other models, many not based on Hill functions, are also possible.

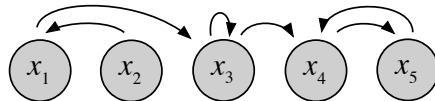


Fig. 8.5

Example GRN with 5 nodes. There is an autoregulation loop at node 3.

8.7.3 Relation between GRN and PRN

Figure 8.5 shows a GRN in the usual biological form, with one node per gene. Figure 8.6 (left) shows the corresponding PRN with just the internal translation arrows from R to P (wavy arrows), which are common to all PRNs. Figure 8.6 (right) shows the transcription arrows (solid) as well. The autoregulation loop on node 3 can be seen as the pair of arrows between nodes x_3^R and x_3^P .

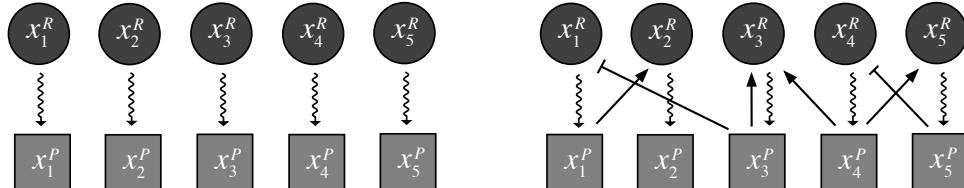


Fig. 8.6

Bipartite structure of PRN for Fig. 8.5. *Left:* translation arrows (wavy) only. *Right:* Typical example with added transcription arrows (solid).

8.7.4 Relation between Jacobian and adjacency matrices

In preparation for the next section, we point out one feature of the relation between adjacency matrices and the Jacobian that might cause confusion. This is a technical issue that arises because different disciplines prefer different conventions. It is basically straightforward, but can cause confusion if it is not borne in mind when reading literature from diverse sources.

The issue is that the natural structure of the Jacobian, and the standard definition in graph theory of an adjacency matrix, do not correspond as well as we might wish. There is one adjacency matrix for each node- or arrow-type. The admissible linear maps are linear combinations of the *transposes* of these matrices. The transpose arises because of different conventions for Jacobian and adjacency matrices.

To explain what we mean, consider the Smolen network of Fig. 8.4. The admissible ODEs are

$$\dot{x}_1 = f(x_1, x_1, x_2) \quad \dot{x}_2 = f(x_2, x_1, x_2). \quad (8.6)$$

Recall that, following most graph theory texts, we adopt the convention that row i of an adjacency matrix represents nodes to which node i sends an *output* arrow. There are three adjacency matrices: one (N) corresponding to the nodes, a second (A) to the activator

arrows, and a third (R) to the repressor arrows. They are:

$$N = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad A = \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix} \quad R = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix}.$$

The Jacobian for (8.6) has the form

$$J = \begin{bmatrix} f_1 + f_2 & f_3 \\ f_2 & f_1 + f_3 \end{bmatrix}$$

where, as explained above, f_k is the partial derivative of f with respect to the k th variable. Now

$$\begin{aligned} J &= f_1 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + f_2 \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix} + f_3 \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix} \\ &= f_1 N^T + f_2 A^T + f_3 R^T. \end{aligned}$$

Thus the Jacobian corresponds naturally to the sum, over the three ‘arrow’ types (node, activator, repressor) of the corresponding partial derivatives of f multiplied by the transpose of the corresponding adjacency matrix. Similar results hold for any network (Golubitsky and Stewart, 2023, Theorem 11.4).

In practice this clash of conventions causes no serious issues, because the Jacobian and its transpose have the same eigenvalues. However, they do not have the same eigenvectors, and it is the eigenvectors of the Jacobian that correspond to important features of the dynamics such as cluster patterns.

The arrangement of entries for the Jacobian is natural because it corresponds to the arrangement of variables in the components of the ODE. The component for \dot{x}_c depends on the variables listed inside the corresponding f , which appear as a row. To avoid the need for a transpose, we can either redefine the Jacobian in an unnatural manner, or redefine the adjacency matrices so that entry ij corresponds to the tails nodes of input arrow to node i , not outputs. Indeed, this is what is done in sources such as (Golubitsky and Stewart, 2023; Pecora and Carroll, 1998; Sorrentino et al., 2016).

8.7.5 Adjacency matrix and Jacobian

The eigenvalues of the Jacobian at a cluster synchronous equilibrium of a PRN can be deduced from those of the corresponding GRN. This simplifies some calculations. The key point is that every PRN is bipartite. The two parts are the mRNA nodes (R) and the protein nodes (P). Arrows from R to P are internal to the combined gene node. Assume there are n genes; then the network has n R nodes and n P nodes.

In this subsection it is convenient to list the nodes with the R nodes as $1, \dots, n$ and the P nodes as $n+1, \dots, 2n$.

The transpose of the weighted adjacency matrix for the network \mathcal{G} , and the Jacobian at a general point, have the form

$$M = \begin{bmatrix} P & Q \\ R & S \end{bmatrix}. \tag{8.7}$$

partitioned into blocks according to the bipartite structure. The blocks P, R, S are diagonal; say

$$P = \begin{bmatrix} p_1 & 0 & \cdots & 0 \\ 0 & p_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & p_n \end{bmatrix} \quad R = \begin{bmatrix} r_1 & 0 & \cdots & 0 \\ 0 & r_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & r_n \end{bmatrix} \quad S = \begin{bmatrix} s_1 & 0 & \cdots & 0 \\ 0 & s_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & s_n \end{bmatrix}$$

In contrast, the block Q is arbitrary, because proteins produced by one gene can regulate mRNA in other genes. In the example of Fig. 8.6 (right),

$$Q = \begin{bmatrix} 0 & q_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ q_2 & 0 & q_3 & 0 & 0 \\ 0 & 0 & q_4 & q_5 & 0 \\ 0 & 0 & 0 & 0 & q_6 \end{bmatrix}.$$

The interpretation of these blocks is:

P : internal mRNA dynamic (usually degradation of mRNA).

Q : interaction matrix Q_{ij} denoting the binding of the transcription factor expressed by gene i to the regulatory domain of gene j to regulate (turn on and off) its expression of mRNA .

R : translation of mRNA to protein.

S : internal protein dynamic (usually degradation of the protein).

The structure of the matrix M in (8.7) does not constrain its eigenvalues in any simple manner. In Section 8.7.5 we specialize to cases where the eigenvalues of M can be determined directly from those of Q , see Theorem 8.6. The matrix Q is the transpose of the adjacency matrix for the GRN version of the network, weighted by connection strengths if appropriate.

Homogeneous case

To make progress we now assume the network is homogeneous. That is, all nodes are input isomorphic. This implies that $p_i = p, r_i = r, s_i = s$ for $1 \leq i \leq n$. Here p, r, s are constants. (For the usual models p, s represent degradation of mRNA and protein concentrations, so are negative, and r represents the rate of translation, so is positive. However, at this stage we allow arbitrary signs.) Now

$$M = \begin{bmatrix} pI_n & Q \\ rI_n & sI_n \end{bmatrix} \tag{8.8}$$

where I_n is the $n \times n$ identity matrix. The eigenvalues of M are related to those of Q , which in general are simpler to compute. We prove:

Theorem 8.6 *If the eigenvalues of Q are μ_i , then those of M are*

$$\lambda_i^\pm = \frac{1}{2} \left(p + s \pm \sqrt{(p - s)^2 + 4r\mu_i} \right). \tag{8.9}$$

Each μ_i gives two values of λ_i , in accordance with doubling the size of the matrix.

Proof Suppose that $\begin{bmatrix} u \\ v \end{bmatrix}$ is an eigenvector of M with eigenvalue λ . Then

$$\begin{bmatrix} \lambda u \\ \lambda v \end{bmatrix} = \begin{bmatrix} pI_n & Q \\ rI_n & sI_n \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} pu + Qv \\ ru + sv \end{bmatrix}.$$

Therefore

$$(\lambda - p)u = Qv \quad (\lambda - s)v = ru$$

so

$$u = \frac{\lambda - s}{r}v \quad \frac{(\lambda - p)(\lambda - s)}{r}v = Qv.$$

Thus v is an eigenvector of Q with eigenvalue

$$\mu = \frac{(\lambda - p)(\lambda - s)}{r}.$$

Rewrite this as

$$(\lambda - p)(\lambda - s) - r\mu = 0 \tag{8.10}$$

whose solutions are (8.9). Now let λ run through all eigenvalues of M . \square

Remark This calculation does not use homogeneity as such—which requires all nodes to be input isomorphic. All it needs is that $p_i = p, r_i = r, s_i = s$ for $1 \leq i \leq n$. That is, degradation and transcription rates are independent of the node, which (unlike homogeneity) does not impose conditions on the block matrix Q . This assumption is often realistic and is common in the literature, in particular in *E. coli* regulation dynamics.

8.8 Stability analysis of biologically relevant circuits

We now consider the GRN circuits in Fig. 8.4. A full discussion can be found in (Stewart et al., 2024), so we present a representative selection of results and refer to that paper for a more extensive analysis. We focus on linear stability of cluster states induced by a fibration.

We also compare the Smolen circuit with the metabolator circuit, which has the same base, hence the same synchronous stability, but different transverse stability.

8.8.1 Stability analysis for lock-on

We begin with the *lock-on* circuit, Fig. 8.7. The figure includes *inputs* I_1, I_2 . For a synchronous state to exist we require $I_1 = I_2 = I$, and we make this assumption. The network then has a group-theoretic symmetry that swaps $x \leftrightarrow y$. Together with the identity permutation, this forms the cyclic group \mathbb{Z}_2 of order 2.

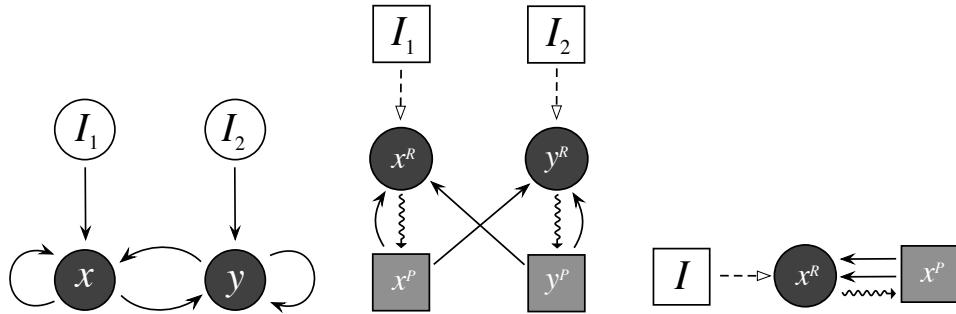


Fig. 8.7

Lock-on circuit, with inputs. *Left:* GRN. *Middle:* PRN. *Right:* Base of fibration when $I_1 = I_2 = I$, shown by colors in Middle and Right figures.

Lock-on: general PRN model

For consistency, we order the component equations as in Section 8.7.2; that is, mRNA nodes first, followed by the corresponding protein nodes. Within each equation, the arguments of the function concerned are listed in an order that respects input isomorphisms. A bar over a set of variables indicates a vertex symmetry: the function is invariant under any permutation of those variables.

Admissible ODEs for the lock-on PRN are:

$$\begin{aligned}\dot{x}^R &= f(x^R, \overline{x^P, y^P}, I) \\ \dot{y}^R &= f(y^R, \overline{x^P, y^P}, I) \\ \dot{x}^P &= g(x^P, x^R) \\ \dot{y}^P &= g(y^P, y^R)\end{aligned}\tag{8.11}$$

in the coordinates $X = (x^R, y^R, x^P, y^P) \in \mathbb{R}^4$. We treat $I \in \mathbb{R}$ as a fixed but arbitrary parameter. We call this the *general model* for the lock-on circuit. In (8.11) the first argument of functions f and g is the respective degradation variable.

The Jacobian for (8.11) is

$$J = \begin{bmatrix} f_1 & 0 & f_2 & f_3 \\ 0 & f_1 & f_2 & f_3 \\ g_2 & 0 & g_1 & 0 \\ 0 & g_2 & 0 & g_1 \end{bmatrix}$$

where, as explained in Section 8.7.4, subscripts f_i, g_i on f and g indicate the partial derivative with respect to the i th variable.

The lock-on circuit has a nontrivial (group-theoretic) fibration indicated by the colors in Fig. 8.7, which shows the orbit coloring for the \mathbb{Z}_2 symmetry. The corresponding synchrony subspace is

$$\Delta = \{(x^R, y^R, x^P, y^P) : x^R = y^R, x^P = y^P\} = \{(u, u, v, v) : u, v \in \mathbb{R}\}.$$

The space Δ can be identified with the state space for the base of the fibration, Fig. 8.7 (right). We take the nodes of the base to be the two colors (blue, red) and identify each point

(u, v) in its state space with (u, u, v, v) in the state space of the lock-on PRN. The restricted ODE on Δ is admissible for the base.

The linear stability of the synchronous state is determined by the eigenvalues of the Jacobian when it is evaluated at a synchronous equilibrium

$$x^R = y^R = u_0 \quad x^P = y^P = v_0$$

which we denote by

$$X_0 = (u_0, u_0, v_0, v_0). \quad (8.12)$$

At such a synchronous equilibrium, vertex symmetry implies that $f_2 = f_3$. This follows from the Taylor series expansion about the equilibrium point concerned. Consider the first term in equation 8.11. The linear part of the Taylor series is $f_1x^R + f_2x^P + f_3y^P$. However, the function is symmetric when we interchange the second and third variables, which gives $f_1x^R + f_2y^P + f_3x^P$. These expressions must be equal, so $f_2 = f_3$ when evaluated at the synchronous state X_0 . (This observation is a special case of a more general result, which we do not address here.)

To simplify the notation further, let

$$a = f_1 \quad b = f_2 = f_3 \quad c = g_1 \quad d = g_2.$$

We emphasize that the entries a, b, c, d are not constants: they are functions of the synchronous equilibrium (8.12). The Jacobian is

$$J = \begin{bmatrix} aI & Q \\ dI & bI \end{bmatrix} \quad \text{where} \quad Q = \begin{bmatrix} b & b \\ b & b \end{bmatrix} \quad (8.13)$$

in accordance with (8.8). (Here I is the 2×2 identity matrix, not an input parameter.) The eigenvalues of Q are $\mu = 0, 2b$. By Theorem 8.6 the eigenvalues of J are:

$$\begin{aligned} \lambda_1 &= a \\ \lambda_2 &= c \\ \lambda_3^\pm &= \frac{1}{2} (a + c \pm \sqrt{K}) \end{aligned} \quad (8.14)$$

where

$$K = (a - c)^2 + 8bd. \quad (8.15)$$

Here λ_1 and λ_2 are real, but λ_3^\pm can be complex, depending on the sign of K . They are real if $K \geq 0$ and a complex conjugate pair if $K < 0$.

Computing the eigenvectors for the eigenvalues λ_3^\pm shows that these eigenvectors lie in the invariant synchrony subspace Δ . Therefore these are the synchronous eigenvalues. In contrast, λ_1, λ_2 have eigenvectors in a complement to Δ and are the transverse eigenvalues.

We claim that the equilibrium X_0 is linearly stable if and only if, when evaluated at X_0 , the partial derivatives a, b, c satisfy one of the following conditions:

- (1) $a < 0, c < 0, a + c + \sqrt{(a - c)^2 + 8bd} > 0, (a - c)^2 + 8bd > 0,$
- (2) $a < 0, c < 0, (a - c)^2 + 8bd < 0.$

The proof is a short series of simple calculations. Let $K = (a - c)^2 + 8bg$ as in (8.15). Linear stability is equivalent to all eigenvalues $\lambda_1, \lambda_2, \lambda_3^+, \lambda_3^-$ having negative real parts. Therefore $a < 0$ and $c < 0$.

If $K > 0$ then λ_3^+ and λ_3^- are real. Since $\lambda_3^+ > \lambda_3^-$, stability requires $\lambda_3^+ < 0$. (The factor $\frac{1}{2}$ in λ_3^\pm does not affect the sign.)

If $K < 0$ then λ_3^+ and λ_3^- are complex conjugate, with real part $\frac{1}{2}(a + c)$. Since $a, c < 0$ this is automatically negative. This proves the claim.

The values of a, b, c, d that can occur in a specific model depend on the model.

Lock-on: special PRN model

For biological applications, the functional forms of the functions f, g in (8.11) are normally chosen from a standard range. A common choice, used here because it is simple and illustrates the analysis, leads to the special model:

$$\begin{aligned}\dot{x}^R &= -\delta x^R + H^+(x^P) + H^+(y^P) + I \\ \dot{y}^R &= -\delta y^R + H^+(x^P) + H^+(y^P) + I \\ \dot{x}^P &= -\alpha x^P + \beta x^R \\ \dot{y}^P &= -\alpha y^P + \beta y^R.\end{aligned}\tag{8.16}$$

The parameters $\alpha, \delta > 0$ represent degradation of mRNA and protein respectively. The parameter $\beta > 0$ is the rate at which the mRNA produces protein. We use the same Hill function H^+ for both activator arrows, since these give identical couplings. This reflects the vertex symmetry of f indicated by the overline.

Consider the synchronous equilibrium X_0 in (8.12), for which $x^R = y^R = u_0, x^P = y^P = v_0$. The quantities u_0 and v_0 depend on the input I , so I can act as a bifurcation parameter.

For (8.16) we have

$$a = -\delta < 0 \quad b = H^{+'}(v_0) > 0 \quad c = -\alpha < 0 \quad d = \beta > 0.$$

where $H^{+'}$ indicates the derivative of H^+ . The sign of b is always positive because H^+ is monotonic increasing, reflecting the activation arrows.

The transverse eigenvalues are $-\delta, -\alpha < 0$, in accordance with the stability criteria (1) and (2) above. Thus the dynamic transverse to the synchrony subspace is attracting; this is caused by the degradation terms. Indeed, here the transverse dynamic is *constant*: independent of the equilibrium point in the synchrony space. The quantity $K = (a - c)^2 + 8bd$ is always positive, so we apply criterion (1): stability also requires $a + c + \sqrt{K} < 0$. This is equivalent to $\sqrt{K} < -a - c$, that is, $K < (a - c)^2$. Therefore $bd < 0$, but this does not occur.

We conclude that the synchronous equilibrium is unstable for this choice of model.

8.8.2 Bistability of the toggle-switch

The toggle-switch is shown in Fig. 8.8. The general model for the toggle-switch is the same as for the lock-on circuit, because the network formalism used to determine admissible

ODEs does not distinguish activator arrows from repressor arrows; and each network has only one type of arrow.

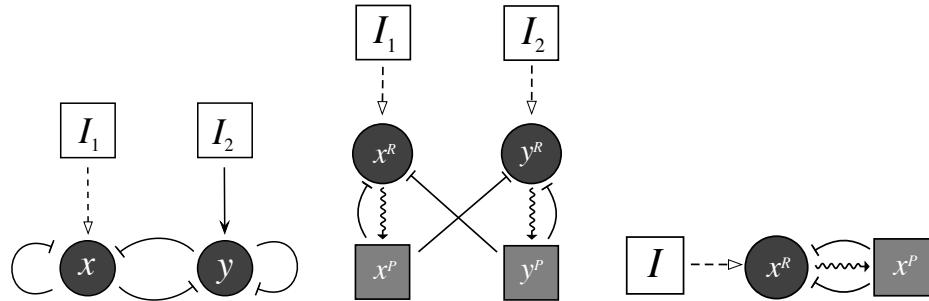


Fig. 8.8

Toggle-switch. *Left:* GRN with inputs. *Middle:* PRN with inputs. *Right:* Base.

Rather than repeating the analysis using a special model analogous to that for the toggle-switch but with H^+ replaced by H^- , we focus on the most interesting feature of the toggle-switch, and the reason for its name. It is *bistable*, and can be switched between two stable equilibria. This is discussed thoroughly in (Gardner et al., 2000), using the following dimensionless model:

$$\dot{u} = \frac{\alpha_1}{1 + v^\beta} - u + I_1 \quad \dot{v} = \frac{\alpha_2}{1 + u^\gamma} - v + I_2. \quad (8.17)$$

Here u, v are the concentrations of the two repressors, the α_i are lumped parameters representing their effective synthesis rates, β is the cooperativity of repression for promoter 2, and γ is the cooperativity of repression for promoter 1. The terms I_1, I_2 are external inputs.

We analyze their model in the case where $\beta = \gamma = 2$ and $I_1 = I_2 = 0$. For notational simplicity we set $\alpha_1 = a, \alpha_2 = b$. There is a fibration symmetry if and only if $\alpha_1 = \alpha_2$ and $\beta = \gamma$, when the symmetry group is \mathbb{Z}_2 . When $\alpha_1 \neq \alpha_2$ we have ‘induced’ symmetry breaking.

The ODE becomes

$$\dot{u} = \frac{a}{1 + v^2} - u \quad \dot{v} = \frac{b}{1 + u^2} - v. \quad (8.18)$$

Restriction to synchrony subspace

We first analyze (8.18) on the synchrony space where $u = v, a = b$, but considering also transverse eigenvalues. The equation becomes

$$\dot{u} = \frac{a}{1 + u^2} - u \quad \text{with equilibria when} \quad u^3 + u = a. \quad (8.19)$$

Since the cubic $u^3 + u$ is monotone increasing and passes through the origin, there is a unique solution $u_0(a)$ for each value of a . This solution is positive when $a > 0$, and $u_0(a)$ increases monotonically with a . We write it as u_0 , remembering that this varies with a .

The Jacobian for (8.19) at u_0 is the 1×1 matrix

$$\hat{J} = \left[-1 - \frac{2au_0}{(1+u_0^2)^2} \right] = -1 - \frac{2u_0^3}{a}$$

since $(1+u_0)^2 = \frac{a}{u_0}$ by (8.19). When $a > 0$ this is always negative, so the equilibrium u_0 is stable with respect to synchrony-preserving perturbations, for all values of a .

The Jacobian for (8.18) is

$$J = \begin{bmatrix} -1 & \frac{-2av}{(1+v^2)^2} \\ \frac{-2bu}{(1+u^2)^2} & -1 \end{bmatrix}$$

Evaluated at a synchronous equilibrium (u_0, u_0) when $a = b$ this becomes

$$J|_{(u_0, u_0)} = \begin{bmatrix} -1 & \frac{-2au_0}{(1+u_0^2)^2} \\ \frac{-2au_0}{(1+u_0^2)^2} & -1 \end{bmatrix}$$

Being a symmetric matrix, this has real eigenvalues. To compute them, we observe that $(1+u_0)^2 = \frac{a}{u_0}$ by (8.19), so

$$J|_{(u_0, u_0)} = \begin{bmatrix} -1 & \frac{-2u_0^3}{a} \\ \frac{-2u_0^3}{a} & -1 \end{bmatrix}$$

The eigenvalues are

$$\lambda_1 = -1 + \frac{2u_0^3}{a} \quad \lambda_2 = -1 - \frac{2u_0^3}{a}$$

with corresponding eigenvectors $[1, -1]^T$ and $[1, 1]^T$. Therefore the transverse eigenvalue is λ_1 and the synchronous eigenvalue is λ_2 . Since $\lambda_1 > \lambda_2$, the first eigenvalue to change sign as u_0 increases is λ_1 , so the synchronous state loses stability when $2u_0^3/a = 1$. Now $a = 2u_0^3 = 2(a - u_0)$, so $u_0 = a/2$, which implies that $(a/2)^3 + (a/2) = a$, whence $a = \pm 2$. The positive value is $a = 2$, and this implies that $u_0 = 1$. Thus the synchronous state is stable in the full state space if $a < 2$, but becomes unstable to synchrony-breaking perturbations when $a > 2$.

Full state space

We now extend the analysis of (8.18) to the full state space and characterize the equilibria and their stabilities. At equilibrium,

$$u = \frac{a}{1+v^2} \quad v = \frac{b}{1+u^2}$$

so

$$u(1+v^2) = a \quad v(1+u^2) = b.$$

Eliminate v using the second equation: the first becomes

$$u \left(1 + \left[\frac{b^2}{(1+u^2)^2} \right] \right) = a.$$

Multiply throughout by $(1 + u^2)^2$:

$$u(1 + u^2)^2 + ub^2 = a(1 + u^2)^2$$

which expands to give the 5th degree polynomial equation

$$p(u) = u^5 - au^4 + 2u^3 - 2au^2 + (b^2 + 1)u - a = 0. \quad (8.20)$$

Plotting the polynomial $p(u)$ numerically for various values of a, b we find that sometimes it has one real zero, and sometimes three real zeros. Fig. 8.9 shows typical plots. Since the curves are close to the horizontal axis near the origin, blow-ups near $u = 0$ clarify the number of zeros.

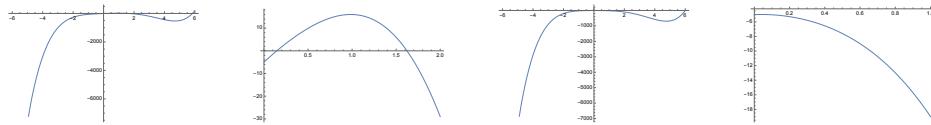


Fig. 8.9

Plots of stability. From left to right: Graph of $p(u)$ when $a = b = 6$. Blowup near $u = 0$ to show two of the three zeros. Graph of $p(u)$ when $a = 6, b = 1$. Blowup near $u = 0$ to show there are no zeros in that region.

Applying a standard viewpoint in singularity/catastrophe theory, we plot the bifurcation variety in (a, b) -space. This is the set of values of (a, b) for which the polynomial has a multiple zero. It divides (a, b) -space into regions where the number of roots is 3 or 1. The condition for a multiple zero is that u is also a zero of the derivative:

$$p'(u) = 5u^4 - 4au^3 - 6u^2 - 4au + (b^2 + 1) = 0. \quad (8.21)$$

Rewrite equations (8.20) and (8.21) as

$$\begin{aligned} a(u^4 + 2u^2 + 1) &= u^5 + 2u^3 + (b^2 + 1)u, \\ a(4u^3 + 4u) &= 5u^4 + 6u^2 + (b^2 + 1). \end{aligned}$$

These are linear in a and b^2 , so we solve explicitly and take the square root of b^2 to get

$$a = \frac{4u^3}{3u^2 - 1}, \quad b = \sqrt{\frac{(u^2 + 1)^3}{3u^2 - 1}}. \quad (8.22)$$

Now we can consider (8.22) as a parametric representation of the *bifurcation variety* in (a, b) -space, using u as parameter—a convenient way to plot the curves. Figure 8.10 (left) shows the result, where we retain only positive values of a and b . Gardner et al. (2000) obtain similar cusped curves for other values of β, γ in (8.17). This figure is symmetric about the diagonal $a = b$, due to the ‘parameter symmetry’ $u \leftrightarrow v, \alpha_1 \leftrightarrow \alpha_2, I_1 \leftrightarrow I_2, \beta \leftrightarrow \gamma$ of (8.17).

The shaded region inside the cusped curve corresponds to a bistable state with two stable equilibria and one unstable one. Outside the cusped curve there is a unique equilibrium, which is stable. The geometry is that of the cusp catastrophe (Poston and Stewart, 2014; Zeeman, 1977); see Fig. 8.10 (right). The cusped curve indicates transitional values of the

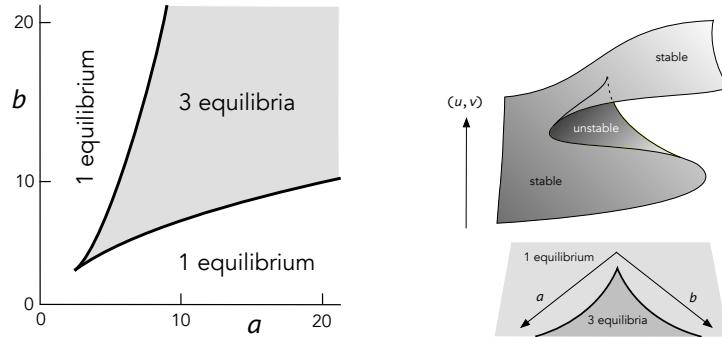


Fig. 8.10

Singularity/catastrophe theory. *Left:* Bifurcation variety in (a, b) -space. *Right:* Cusp catastrophe surface showing how equilibria vary with parameters (a, b) . (Schematic to show topology: the vertical axis is 2-dimensional.)

parameters (α_1, α_2) at which a stable equilibrium merges with an unstable one. With quasi-static dynamics (the equilibrium state changes continuously as parameters vary, except for discontinuous jumps when that state disappears) the system jumps between the upper and lower sheets of the surface at the folds, and thus exhibits hysteresis, leading to the characteristic toggle-switch behavior of this circuit.

The intersection of the diagonal line $\alpha_1 = \alpha_2$ with the cusp catastrophe surface gives a pitchfork bifurcation: see Section 8.9. This is symmetric under the map $(u, v) \mapsto (v, u)$ but appears asymmetric if projected onto u or v . There are intervals of uni- and bi-stability but no hysteresis. Hysteresis requires asymmetric parameter values.

8.8.3 Comparison of metabolator and Smolen

To illustrate how the transverse eigenvalues affect the stability of a synchronous state, when this state is stable on the synchrony subspace, we consider the Smolen circuit and the similar but subtly different metabolator circuit, Fig. 8.11.



Fig. 8.11

Sample circuits with the same base. *Left:* Metabolator network. *Right:* Smolen circuit.

In this section we use a ‘lumped’ model with a single variable for each node, rather than the two variables of a PRN model, since this illustrates the main point in a simple manner. The node variables are then $x_1, x_2 \in \mathbb{R}$ for each circuit.

The admissible ODEs for the metabolator are

$$\begin{aligned}\dot{x}_1 &= f(x_1, x_1, x_2) \\ \dot{x}_2 &= f(x_2, x_2, x_1)\end{aligned}\tag{8.23}$$

where the function f is arbitrary. The variables inside f are taken in the order: node, repressor arrow, activator arrow.

For the Smolen circuit, the admissible ODEs are

$$\begin{aligned}\dot{x}_1 &= f(x_1, x_1, x_2), \\ \dot{x}_2 &= f(x_2, x_1, x_2).\end{aligned}\tag{8.24}$$

If we set $x_1 = x_2 = x$, a balanced synchrony pattern, both ODEs reduce to

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

Using the same f is both models, they have identical synchronous dynamics. However, the stability of the synchronous state to synchrony-breaking perturbations—transverse stability—can be different.

Assume that (8.23) has a synchronous equilibrium point at $x^* = (x_0, x_0)$ for some $y_0 \in \mathbb{R}$. Then (8.24) also has a synchronous equilibrium point at x^* , since the equations are identical. Such an equilibrium is linearly stable if all eigenvalues of the Jacobian at x^* have negative real parts. By the chain rule, the Jacobians J_M for the metabolator and J_S for the Smolen network are

$$\begin{aligned}J_M &= \begin{bmatrix} \partial_1 f + \partial_2 f & \partial_3 f \\ \partial_3 f & \partial_1 f + \partial_2 f \end{bmatrix}_{x^*} = \begin{bmatrix} \alpha + \beta & \gamma \\ \gamma & \alpha + \beta \end{bmatrix} \\ J_S &= \begin{bmatrix} \partial_1 f + \partial_2 f & \partial_3 f \\ \partial_2 f & \partial_1 f + \partial_3 f \end{bmatrix}_{x^*} = \begin{bmatrix} \alpha + \beta & \gamma \\ \beta & \alpha + \gamma \end{bmatrix}\end{aligned}$$

(say), where the subscript indicates evaluation at x^* . The corresponding eigenvalues are easily seen to be

$$\begin{aligned}\text{metabolator: } &\alpha + \beta + \gamma & \alpha + \beta - \gamma, \\ \text{Smolen: } &\alpha + \beta + \gamma & \alpha.\end{aligned}$$

The condition for stability to synchrony-preserving perturbations is $\alpha + \beta + \gamma < 0$, which is the same for both networks. This is the synchronous eigenvalue.

Assuming stability to synchrony-preserving perturbations, an additional condition for stability to synchrony-breaking perturbations is required. This is the transverse eigenvalue

$$\begin{aligned}\text{metabolator: } &\alpha + \beta - \gamma < 0, \\ \text{Smolen: } &\alpha < 0,\end{aligned}$$

and these conditions are different. Figure 8.12 sketches the regions of the (α, γ) plane in which various stability conditions hold, for $\beta < 0$, $\beta = 0$, and $\beta > 0$. The regions are as follows:

P: Stable to synchrony-preserving perturbations (same condition for metabolator and Smolen).

M: Satisfying P, and also stable to synchrony-breaking perturbations (hence stable) for metabolator.

S: Satisfying P, and also stable to synchrony-breaking perturbations (hence stable) for Smolen.

For suitable choices of α, β, γ , it is possible for both networks, either one, or neither to be stable. Both can be stable or both can be unstable to synchrony-preserving perturbations.

It is not possible for one to be stable to synchrony-preserving perturbations while the other is unstable to synchrony-preserving perturbations, since the restricted ODE is the same in both cases.

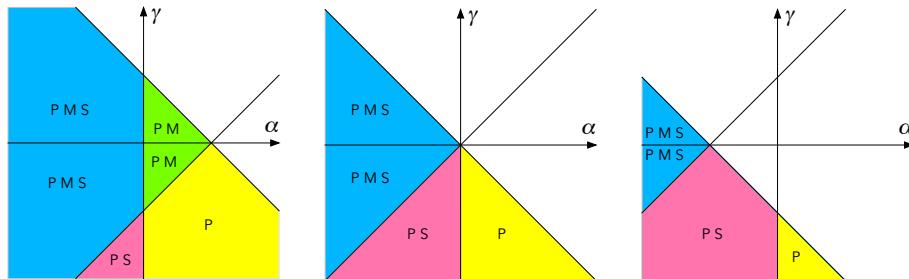


Fig. 8.12

Stability regions in the (α, γ) plane. P = both metabolator and Smolen stable to synchrony-preserving perturbations; M = metabolator also stable to synchrony-breaking perturbations; S = Smolen network also stable to synchrony-breaking perturbations. *Left:* $\beta < 0$. *Middle:* $\beta = 0$. *Right:* $\beta > 0$.

In general the stability of synchrony patterns depends in complicated ways on the model ODE, especially for non-equilibrium states. There are few useful general principles, although some results exist for regular, symmetric, and feed-forward networks (Golubitsky and Stewart, 2023, Chapter 18), or with mild extra assumptions on the model ODE (Huang et al., 2009).

8.9 Bifurcations—a short sketch and examples

We now move on from specific networks and models to give a brief summary of the commonest bifurcations. These are the ‘local’ bifurcations, where every important change happens in any sufficiently small neighborhood of the bifurcation point.

In a one-parameter family of ODEs, which is common in applications, it is possible to track how a given state changes as the parameter λ varies. For ‘most’ parameter values, this state usually varies continuously with the parameter and retains its dynamical character—steady, periodic, chaotic, and so on. However, there can exist *bifurcation points* $\lambda = \lambda_0$ at which the type of the state, or the number of possible states, changes. The system then adopts a different state. Such a change is called a *bifurcation*. The main cause of bifurcation is *loss of stability* of the state being tracked. Typically, the values of λ at which a bifurcation occurs are isolated.

Bifurcations provide a powerful method for analyzing stabilities, because they split the parameter values into intervals along which the stability remains the same. (When there is more than one parameter, similar ideas apply, but now bifurcations typically occur on ‘codimension 1’ subsets of parameter space—that is on curves for two parameters, surfaces for three, and dimension $d - 1$ submanifolds for d parameters.) In Fig. 8.12 the straight

lines bounding the various regions are sections of planes, and these planes define values of (α, β, γ) at which bifurcations from synchronous equilibrium occur. In general these ‘bifurcation varieties’ can be curved.

In dynamical systems theory there are many standard ‘generic’ bifurcation scenarios (Guckenheimer and Holmes, 1983). Networks introduce new complexities, because behavior that is generic in a general dynamical system need not be generic in a network dynamical system. More precisely, the structure of an admissible ODE for a network has a strong influence on the possible bifurcations. The relation between the possible bifurcations and the graph topology is, in general, subtle and complex (Leite and Golubitsky, 2006). However, a few useful general principles apply (Golubitsky and Stewart, 2023, Chapters 3,6, 18–22).

Definition 8.7 Bifurcation. A *bifurcation* occurs in a λ -parametrized family of ODEs $\dot{x} = f(x, \lambda)$ when the set of attractors changes its topological type as the parameter $\lambda \in \mathbb{R}^n$ varies. The parameter can be multidimensional ($n > 1$) but is commonly 1-dimensional ($\lambda \in \mathbb{R}$).

Local bifurcation occurs when a parametrized family of equilibria loses stability, leading to one or more branches of new solutions.

An eigenvalue is *critical* if it has zero real part.

There are two kinds of generic local bifurcation:

- (a) *Steady-state* bifurcation: a real eigenvalue of the Jacobian (linearization) passes through 0 as λ passes through some value λ_0 . The generic bifurcation is saddle-node for a general dynamical system, transcritical if the family has a branch of trivial solutions, and pitchfork if the system has a symmetry of order 2. Pitchforks and higher singularities can occur generically if $n > 1$, and symmetry is not essential (Golubitsky et al., 1988).
- (b) *Hopf* bifurcation: a pair of complex conjugate eigenvalues of the Jacobian crosses the imaginary axis as λ passes through some value λ_0 . Typically this causes a new branch of time-periodic states, with frequency tending to $\frac{2\pi}{\omega}$ at the bifurcation point, where the imaginary eigenvalues are $\pm\omega i$. See (Hassard et al., 1981).

In either case, λ_0 is the *bifurcation point* concerned.

Figure 8.13 shows bifurcation diagrams for these two types of local bifurcation from a branch of stable equilibria, and variants of them under various extra conditions. (a) For arbitrary f the only generic local bifurcation to an equilibrium state from a branch of equilibria is the saddle-node, where the stable equilibrium merges with an unstable one and no equilibria exist for $\lambda > 0$. (b) If $f(0, \lambda) = 0$ for all λ , a common assumption in network models, a transcritical bifurcation (‘exchange of stability’) is generic. (c,d) If $f(x, \lambda)$ is even in x the generic bifurcation is a pitchfork, which can be supercritical or subcritical. (e,f) For arbitrary f the other generic local bifurcation from a branch of equilibria is Hopf bifurcation to a branch of periodic states, which can be supercritical or subcritical. (g) A subcritical Hopf branch can ‘turn around’ and become stable.

In a network with group symmetry, bifurcations can either preserve symmetry or break it. *Symmetry breaking* bifurcations are of special interest because they lead to a change of symmetry, which manifests as pattern formation. Similarly, in a network with fibration

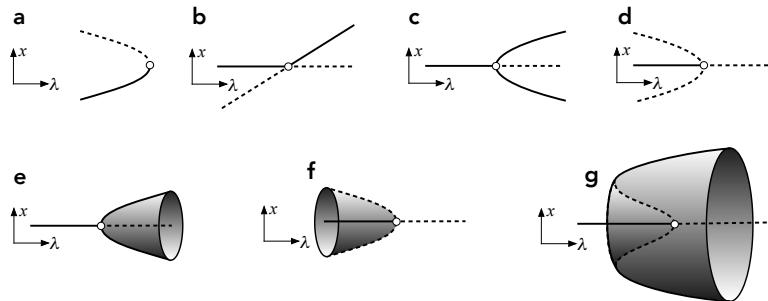


Fig. 8.13

Standard types of bifurcation. Solid line: stable. Dashed line: unstable. White dot: bifurcation point. (a) Saddle-node. (b) Transcritical, generic when $f(0, \lambda) = 0$ for all λ . (c) Supercritical pitchfork, generic when $f(x, \lambda)$ is even in x . (d) Subcritical pitchfork, generic when $f(x, \lambda)$ is even in x . (e) Supercritical Hopf bifurcation. (f) Subcritical Hopf bifurcation. (g) Subcritical Hopf bifurcation ‘turning around’ to create a stable branch, not local to the origin.

symmetry, bifurcations can either preserve synchrony or break it. *Synchrony-breaking* bifurcations are of special interest because they lead to a change of the synchrony pattern, which again can manifest as pattern formation. The two bifurcations set-ups are analogous, and conform to the group symmetry/fibration distinction.

An informative example occurs for the toggle-switch circuit, Section 8.8.2, which has \mathbb{Z}_2 symmetry swapping the two nodes. We observed that the synchronous state loses stability to synchrony-breaking perturbations (but not synchrony-preserving ones) when the parameters satisfy $a = b = 2$. These values correspond to the cusp point in Fig. 8.10. The cross-section of the cusp catastrophe surface along the diagonal $a = b$ is a pitchfork bifurcation in (u, v) -space.

Hopf bifurcation for the Smolen and metabolator circuits is discussed in (Stewart et al., 2024).

8.9.1 First bifurcation

One useful general stability principle is the notion of ‘first bifurcation’. For an ODE on \mathbb{R}^k where $k > 1$, the Jacobian J has more than one eigenvalue. Eigenvalues move continuously as parameters vary continuously (Lancaster and Tismenetsky, 1985). Bifurcations occur when one of these eigenvalues crosses the imaginary axis.

A standard way to analyze bifurcations is to arrange for the bifurcation parameter λ appear in the ODE as the family $\dot{x} = F(x) + \lambda x$. In this case, all eigenvalues move towards the right in \mathbb{C} as λ increases. Therefore stability of the equilibrium concerned is first lost when the eigenvalue with largest real part crosses the imaginary axis; we call this the *first bifurcation*. A similar remark applies if the bifurcation parameter appears as an added input, such as I_1, I_2 in (8.17), since these have no effect in the eigenvalues because the form of

the Jacobian does not change. However, the point at which it is evaluated depends on the value(s) of the input(s).

If the path of equilibria persists for larger values of λ another bifurcation from that path occurs if another eigenvalue crosses the imaginary axis. However, the path of equilibria leading up to that bifurcation is unstable, because at least one eigenvalue now has positive real part, so the bifurcating branch initially describes unstable states. Therefore the most significant bifurcation for applications is the first one.

That said, other choices of bifurcation parameter can change the ordering of the real parts of eigenvalues, because the form $\dot{x} = F(x) + \lambda x$ is not the only possible choice. The family of ODEs must remain admissible, but μx can be replaced by an independent linear term for each type of node and each type of arrow, or by nonlinear expressions. We omit details.

Stewart et al. (2024) tabulates the conditions for first bifurcation for the six circuits featured in Fig. 8.4, for the general admissible PRN model, on the assumption that the bifurcation parameter arises as an added term μx or as an added input. We omit details here because they depend on various partial derivatives, and the notation used depends on the circuit. Table 8.1 provides a summary stating the type of the first bifurcation, again under the stated assumption on how the bifurcation parameter appears in the model.

circuit	possible type of first bifurcation		
lock-on	SB steady	SP steady	
toggle-switch	SB steady	SP steady	
Smolen oscillator	SB steady	SP steady	
feed-forward fiber (FFF)	SB steady	SP steady	
Fibonacci fiber	SB steady	SP steady	SP Hopf
Repressilator	SB Hopf	$\frac{1}{3}$ -period phase relation	rotating wave

Table 8.1: Possible first bifurcations from a stable synchronous equilibrium in the PRN model, for the six circuits. SB = synchrony-breaking, SP= synchrony-preserving.

Example 17 First bifurcation for the lock-on circuit As an example of how Table 8.1 is derived, we study the conditions for first bifurcation in the lock-on circuit.

Routine calculations using (8.14) determine which eigenvalues can occur as the first bifurcation. The idea is to compare the real parts of pairs of eigenvalues and determine when they are equal: this happens when they change order. Let $K = (a - c)^2 + 8bd$. For λ_3^\pm we must again distinguish two cases: $K < 0$, where λ_3^\pm are complex conjugate, so Hopf bifurcation may occur, and $K > 0$, where λ_3^\pm are real, so only steady-state bifurcation is possible. We summarize the results in Table 8.2.

Case 1: $K < 0$.

Subcase 1: λ_1 critical.

Here $a = 0$. We have

$$\lambda_2 = c \quad \lambda_3^\pm = \frac{1}{2}(c \pm \sqrt{c^2 + 8bd})$$

eigenvalue	$K < 0$	$K \geq 0$	type of bifurcation
μ_1	$a = 0, c < 0$	$a = 0, c < 0, bd < 0$	SB steady SB steady
μ_2	$a < 0, c = 0$	$a < 0, c = 0, bd < 0$	SB steady SB steady
μ_3	impossible	$a < 0, c < 0, ac = 2bd$	SP Hopf SP steady

Table 8.2: Conditions for first bifurcation from a synchronous equilibrium for the general PRN lock-on and toggle-switch models. SB = synchrony-breaking, SP= synchrony-preserving.

All three of these have real part with the same sign as c . Therefore when $K < 0$, the first bifurcation occurs at μ_1 if and only if $a = 0, c < 0$.

Subcase 2: λ_2 critical.

Here $c = 0$. We have

$$\lambda_1 = a \quad \lambda_3^\pm = \frac{1}{2}(a \pm \sqrt{a^2 + 8bd})$$

All three of these have real part with the same sign as a . Therefore when $K < 0$, the first bifurcation occurs at μ_2 if and only if $a < 0, c = 0$.

Subcase 3: μ_3^\pm critical. (Being complex conjugates, both eigenvalues are critical simultaneously.)

Here $a + c = 0$ so $c = -a$. Now

$$\lambda_1 = a \quad \lambda_2 = -a$$

which cannot both be negative. Therefore μ_3^\pm cannot be the first bifurcation.

Case 2: $K > 0$.

This condition is equivalent to

$$bd > -\frac{(a - c)^2}{8} \tag{8.25}$$

Subcase 1: μ_1 critical.

Now $a = 0$ and

$$\lambda_2 = c \quad \lambda_3^\pm = \frac{1}{2}(c \pm \sqrt{c^2 + 8bd})$$

Thus $c < 0$ and $c + \sqrt{c^2 + 8bd} < 0$. Since $c < 0$ this is easily seen to be equivalent to $bd < 0$.

Subcase 2: λ_2 critical.

Now $c = 0$, and

$$\lambda_2 = c \quad \lambda_3^\pm = \frac{1}{2}(a \pm \sqrt{a^2 + 8bd})$$

Therefore $a < 0$. A similar calculation again leads to $bd < 0$.

Subcase 3: λ_3^\pm critical. Since $\lambda_3^- < \lambda_3^+$, first bifurcation must happen for λ_3^+ , if at all.

We have $a + c + \sqrt{K} = 0$, which by a simple calculation is equivalent to

$$ac = 2bd$$

Now we want $\lambda_1 = a < 0$ and $\lambda_2 = c < 0$. Therefore $ac > 0$, so $bd > \frac{1}{2}ac$. This is stronger than (8.25).

Example 18 In some cases, certain eigenvalues cannot occur as the first bifurcation, no matter how the bifurcation parameter is chosen. An example occurs for the Smolen circuit, whose general model has the form

$$\begin{aligned}\dot{x}^R &= f(x^R, x^P, y^P, I) \\ \dot{y}^R &= f(y^R, x^P, y^P, I) \\ \dot{x}^P &= g(x^P, x^R) \\ \dot{y}^P &= g(y^P, y^R)\end{aligned}$$

There is a nontrivial fibration symmetry $x = y$, giving a flow-invariant synchrony subspace. At a synchronous equilibrium define

$$a = f_1 \quad b = f_2 \quad c = g_1 \quad d = g_2 \quad e = f_3$$

The Jacobian is

$$J = \begin{bmatrix} aI & Q \\ dI & cI \end{bmatrix} \quad \text{where} \quad Q = \begin{bmatrix} b & e \\ b & e \end{bmatrix}$$

The eigenvalues of Q are $\mu = 0, b + e$. By Theorem 8.6 the eigenvalues of J are

$$\lambda_1 = a \quad \lambda_2 = c \quad \lambda_3^\pm = \frac{1}{2} \left(a + c \pm \sqrt{K} \right)$$

where $K = (a - c)^2 + 4d(b + e)$. Of these, λ_1, λ_2 are synchrony-breaking. The λ_3^\pm pair is synchrony-preserving. Hopf bifurcation can occur only for λ_3^\pm , when $K < 0$. The real part of λ_3^\pm is $\frac{1}{2}(a + c)$, the average of the other two eigenvalues a and c . A Hopf branch can arise only when $\frac{1}{2}(a + c) = 0$, but a necessary condition for this branch to be stable is that the other two eigenvalues a and c are negative. This combination is impossible.

Thus the Smolen circuit with fibration symmetry cannot exhibit stable oscillations created at a stable synchrony-breaking Hopf bifurcation. However, stable oscillations can occur through other dynamic mechanisms. Hasty et al. (2002) and Stricker et al. (2008) analyze the Smolen circuit using different mathematical models, and find oscillations that arise by Hopf bifurcation—but not by a synchrony-breaking bifurcation from a synchronous state. Indeed, those models do not have a synchronous state; they employ asymmetric parameter values and input values.

We conclude that more complex models than those discussed above are required to generate sustained oscillations in the Smolen circuit. Indeed, even in the cited models, the parameter regions for Hopf bifurcation are relatively small and delicate. Thus the presence of oscillations in models of the Smolen circuit is highly sensitive to the functions used.

8.9.2 Hopf bifurcation in the symmetric repressilator

We illustrate the effect of symmetric network topology on Hopf bifurcation with a discussion of the *repressilator* (Elowitz and Leibler, 2000). The repressilator does not occur naturally in *E. coli*, but its base is in the class 0-FFF as shown in Fig. 8.4. In (Elowitz and Leibler, 2000) it is modeled both in the symmetric case (where it has \mathbb{Z}_3 symmetry) and in an asymmetric generalization. The symmetric case exhibits sustained periodic oscillations in which successive genes are $\frac{1}{3}$ -period out of phase. In the asymmetric case the oscillations

persist, with phase shifts near $\frac{1}{3}$ -period, but varying amplitudes (partly due to including stochastic noise in the model).

Here we derive analytic results for the general PRN model in the \mathbb{Z}_3 -symmetric case, Fig. 8.14. We show that the $\frac{1}{3}$ -period phase shifts occur through a symmetry breaking Hopf bifurcation, and that this must be the first bifurcation. We also perform simulations that illustrate this phase pattern in a specific model.

The mRNA and protein variables are $x = (x^R, x^P)$, $y = (y^R, y^P)$, $z = (z^R, z^P)$. Admissible ODEs are:

$$\begin{aligned}\dot{x}^R &= f(x^R, z^P) \\ \dot{x}^P &= g(x^P, x^R) \\ \dot{y}^R &= f(y^R, x^P) \\ \dot{y}^P &= g(y^P, y^R) \\ \dot{z}^R &= f(x^R, y^P) \\ \dot{z}^P &= g(z^P, z^R)\end{aligned}$$

The Jacobian J is block-circulant, because of \mathbb{Z}_3 symmetry. Let

$$A = \begin{bmatrix} a & 0 \\ c & d \end{bmatrix} \quad B = \begin{bmatrix} 0 & b \\ 0 & 0 \end{bmatrix}$$

where

$$a = f_1 \quad b = f_2 \quad c = g_2 \quad d = g_1$$

Then in block matrix notation

$$J = \begin{bmatrix} A & 0 & B \\ B & A & 0 \\ 0 & B & A \end{bmatrix}$$

Using representation theory and equivariant dynamics (Golubitsky et al., 1988), or by direct

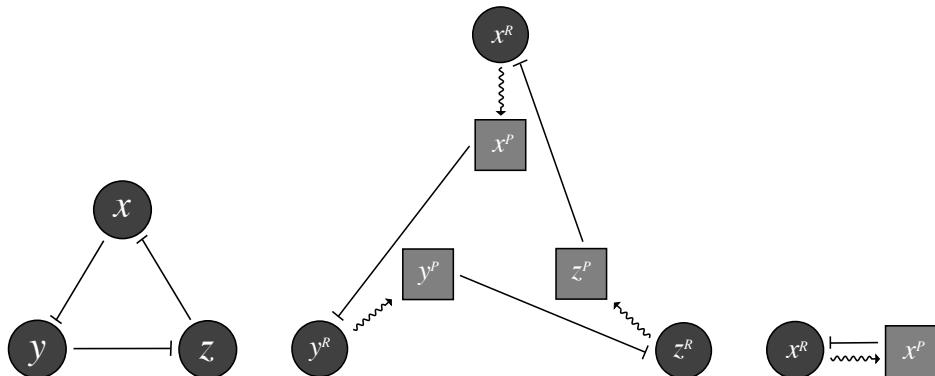


Fig. 8.14

Repressilator circuit. *Left:* GRN. *Middle:* PRN, drawn to show \mathbb{Z}_3 symmetry. *Right:* Base of PRN for (blue, red) fibration symmetry.

calculation, the eigenvalues and form of the eigenvectors of J are:

$$\begin{aligned} \text{eigenvalues of } A + B &: [u, u, u]^T \\ \text{eigenvalues of } A + \zeta^2 B &: [u, \zeta u, \zeta^2 u]^T \\ \text{eigenvalues of } A + \zeta B &: [u, \zeta^2 u, \zeta u]^T \end{aligned}$$

where $u \in \mathbb{R}^2$ is an eigenvector of the 2×2 matrix listed and $\zeta = e^{2\pi i/3}$ is a primitive cube root of unity. Here $A + B$ is symmetry-preserving and the others are symmetry breaking. Now

$$A + B = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \quad A + \zeta B = \begin{bmatrix} a & \zeta b \\ c & d \end{bmatrix} \quad A + \zeta^2 B = \begin{bmatrix} a & \zeta^2 b \\ c & d \end{bmatrix}$$

For Hopf bifurcation we seek purely imaginary eigenvalues of these matrices.

We claim that if the first bifurcation is Hopf, it must be symmetry breaking. That is, the nodes do not oscillate in synchrony, with the same phases. To see why, suppose, on the contrary, that the bifurcation is symmetry-preserving. Then it comes from a pair of imaginary eigenvalues of $A + B$, so the trace $a + d = 0$ and $d = -a$. The eigenvalues of $A + \zeta B$ sum to give the trace of $A + \zeta B$, but again this is $a + d = 0$. Therefore the real parts of these eigenvalues also sum to zero, so one is positive and the other negative. Therefore the first bifurcation cannot arise from $A + B$.

Thus it must be a symmetry breaking Hopf bifurcation, arising either from $A + \zeta B$ or $A + \zeta^2 B$. Since these are complex conjugates, we can consider just $A + \zeta B$. The conditions for $A + \zeta B$ to have purely imaginary eigenvalues can be derived without difficulty, and are:

$$ad + \frac{1}{2}bc > 0 \tag{8.26}$$

$$(a+d)^2(ad + \frac{1}{2}bc) = \frac{3}{4}b^2c^2 \tag{8.27}$$

The way ζ appears in the eigenvectors of J exhibits the $\frac{1}{3}$ -period phase shift pattern, since multiplication by ζ rotates the complex plane through 120° counterclockwise. The Equivariant Hopf Theorem (Golubitsky et al., 1988, Chapter XVI Theorem 4.1) proves that this pattern persists, exactly, in the bifurcating periodic solution of the nonlinear ODE. Thus a symmetry breaking Hopf bifurcation corresponds to the repressor's rotating wave state with $\frac{1}{3}$ -period phase shifts, simulated below in Fig. 8.15.

We consider a numerical example for the following model:

$$\begin{aligned} f(x^R, x^P) &= -\delta x^R + H^-(x^P) \\ g(x^P, x^R) &= -\alpha x^P + \beta x^R \end{aligned}$$

Figure 8.15 shows typical time series for all six variables, superposed in threes: mRNA on the left, protein on the right. It can be checked that this state arises by symmetry breaking Hopf bifurcation. The oscillations are very similar to those in (Elowitz and Leibler, 2000), having the $\frac{1}{3}$ -period phase shift rotating wave form predicted by equivariant bifurcation theory (Golubitsky et al., 1988).

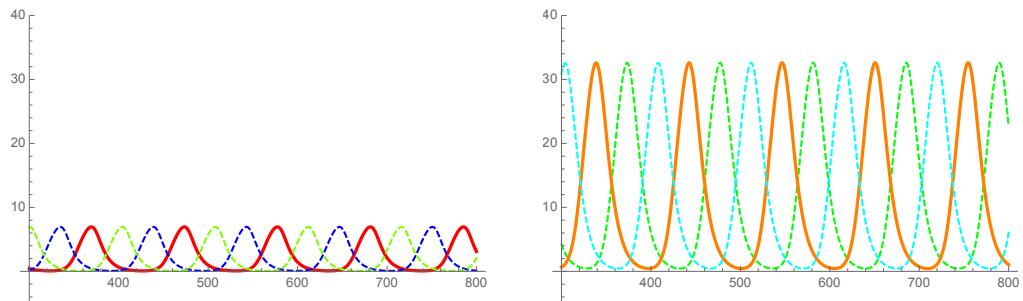


Fig. 8.15

Simulations of special PRN model for repressilator. We use $S(x) = 1/(1+x^2)$ and parameters: $\alpha = 0.2$, $\beta = 1$, $\delta = 0.1$. *Left:* Time series of x^R, y^R, z^R superposed. *Right:* Time series of x^P, y^P, z^P superposed.

8.9.3 Discussion and Conclusions

In the previous sections we analyzed six circuits that serve as functional building blocks in natural or synthetic gene regulatory networks, classified into classes based on their fibration symmetry. We emphasized analytic conditions for local bifurcation from synchronous states in each circuit, focusing on fibration symmetries and synchrony-breaking local bifurcations. The main results are summarized in Table 8.1.

The lock-on, toggle-switch, Smolen circuit, feed-forward fiber, and Fibonacci fiber support both steady symmetry breaking and symmetry preserving bifurcations. Additionally, the Repressilator supports robust oscillatory states via symmetry-breaking Hopf bifurcation, taking the form of rotating waves with $\frac{1}{3}$ -period phase shifts. The Fibonacci fiber circuit supports damped oscillatory states. Oscillations in the Smolen circuit cannot arise by synchrony-breaking Hopf bifurcation in a GRN model because of sign constraints, but can occur in neuron models and through other mechanisms. This analysis enhances the understanding of gene regulatory networks and highlights the importance of both group-theoretic and fibration symmetry for their dynamics. Overall, this study contributes to a systematic characterization of the building blocks of bacterial regulatory networks. It complements previous studies based only on topological properties of the circuits (Leifer et al., 2020; Morone et al., 2020) by extending the analysis to the stability properties of the dynamical solutions observed for biologically inspired ODEs that are admissible for the graphs.

8.10 Master stability function for complete synchronization

Pecora and Carroll (1998) introduced the ‘master stability function’ as a way to study synchronization of a completely synchronous state. They work with Laplacian-based models,

which we discussed briefly in Section 3.6.1. Their approach is described next. Its main disadvantage is that it assumes specific types of couplings (diffusive), which precludes its applicability to many biological networks. The fact that the complete synchronous state is built into the model from the start is also not realistic in biological situations since this state is deleterious for biological functionality. However, when the form of equations that it assumes is an appropriate model it has three fundamental advantages:

- (1) It allows the study of arbitrary synchronous solutions, possibly periodic or chaotic.
- (2) It permits the analysis of stability of the complete synchronous solution for arbitrary networks described by possibly weighted adjacency matrices.
- (3) It leads to a dimensionality reduction of the stability problem into lower dimensional ‘modes’.

The latter point is of relevance to networks with a large number of nodes. It also leads to another indirect benefit: the stability problem can be parametrized in a convenient form, as we see in what follows.

In this approach to stability, the model ODE is assumed to have a specific structure, which, among other things, permits the existence of a completely synchronous state for any structure of the network, whether it has symmetries or not (see section 3.6.1). The assumption is:

$$\dot{x}_i = F(x_i) - S \sum_{j \neq i} A_{ij}(H(x_i) - H(x_j)), \quad i = 1, \dots, N \quad (8.28)$$

for suitable functions F, H , possibly weighted adjacency matrix $A = \{A_{ij} \geq 0\}$, and coupling strength parameter $S > 0$. The state of node i is $n \geq 1$ dimensional, and the number of nodes N is at least 2. The same function F is used for all variables; that is, the internal dynamic is assumed to be the same for all nodes. The coupling terms are of generalized diffusive type; in particular, the coupling between two nodes vanishes whenever they are in the same state. However, it need not vanish when node states differ.

Equation (8.28) can be transformed into a convenient form by defining the *network Laplacian* $L = \{L_{ij}\}$, with entries $L_{ij} = \delta_{ij}(\sum_j A_{ij}) - A_{ij}$, where δ_{ij} is the Kronecker delta. In Laplacian notation, Equation (8.28) can be rewritten as follows,

$$\dot{x}_i = F(x_i) - S \sum_{j \neq i} L_{ij}H(x_j) \quad i = 1, \dots, N. \quad (8.29)$$

By construction, all the rows of the Laplacian matrix $L = \{L_{ij}\}$ sum to zero, which implies that $[1, 1, \dots, 1]$ is an eigenvector with eigenvalue $\lambda_1 = 0$. This ensures the existence of a completely synchronous solution

$$x_1(t) = x_2(t) = \dots = x_N(t) = x_s(t) \quad (8.30)$$

for all t . This solution obeys the equation

$$\dot{x}_s(t) = F(x_s(t)). \quad (8.31)$$

Typically we are interested in those models (8.31) that produce either periodic or chaotic oscillations.

From the viewpoint of modern dynamical systems, an equilibrium is stable if and only

if all eigenvalues of the Jacobian have negative real part at the equilibrium point, see Definition 8.2. In the literature on the master stability function, this leads to a more specific description, which can also be used to study stability of more complex dynamical states. We therefore present the ideas involved in that form.

To study stability we consider small perturbations $\delta x_i(t) = (x_i(t) - x_s(t))$ about the synchronous solution, each of which evolves according to the variational equation

$$\delta \dot{x}_i(t) = DF(x_s(t))\delta x_i(t) - S \sum_j L_{ij} DH(x_s(t))\delta x_j(t) \quad i = 1, \dots, N. \quad (8.32)$$

Here $DF(x_s(t))$ is the Jacobian of F and $DH(x_s(t))$ is the Jacobian of H , both evaluated at the synchronous solution. Equation (8.32) can also be rewritten in vector form

$$\delta \dot{X}(t) = [I_N \otimes DF(x_s(t)) - SL \otimes DH(x_s(t))] \delta X(t) \quad (8.33)$$

where $\delta X(t) = [\delta x_1^T(t), \delta x_2^T(t), \dots, \delta x_N^T(t)]^T$ is an $N \times n$ dimensional vector, and the symbol \otimes indicates the Kronecker product (or tensor product) of matrices.

If L is diagonalizable, the linearization (8.33) decomposes into components along the eigenvectors of L , and the corresponding solutions are the following decoupled ‘modes’.

To see this, we apply the transformation $V^{-1} \otimes I_n$ to (8.33), where V is the matrix whose columns are the eigenvectors of L , $V^{-1}LV = \Lambda$, Λ is a diagonal matrix that has the eigenvalues of L on the main diagonal, and I_n is the identity matrix of dimension n . Now

$$\dot{Y}(t) = [I_N \otimes DF(x_s(t)) - S\Lambda \otimes DH(x_s(t))]Y(t) \quad (8.34)$$

where $Y(t) = V^{-1} \otimes I_n \delta X(t)$. This equation decouples as

$$\dot{\eta}_k(t) = [DF(x_s(t)) - S\lambda_k DH(x_s(t))] \eta_k(t) \quad (8.35)$$

$k = 1, \dots, N$, where $\eta_k(t)$ is the eigenmode associated with the eigenvalue λ_k of L , $k = 1, \dots, N$. The modes (8.35) are independent of one another. One mode (the one associated with the eigenvalue $\lambda_1 = 0$ of L) corresponds to a perturbation inside the synchrony subspace with associated eigenvector $[1, 1, \dots, 1]$. Such a mode is synchrony-preserving. The remaining modes associated with the ‘transverse eigenvalues’ of L , namely $\lambda_2, \lambda_3, \dots, \lambda_N$, correspond to perturbations in the transverse subspace. These modes are synchrony-breaking. We emphasize that based on the assumption that $A_{ij} \geq 0$, the minus sign on S means that the real parts ρ of the eigenvalues $\lambda_k = \rho_k + i\sigma_k$ must all be positive.

We can rewrite (8.35) in parametric form

$$\dot{\eta}_k(t) = [DF(x_s(t)) + \xi DH(x_s(t))] \eta_k(t) \quad (8.36)$$

where the parameter ξ may be complex. Equation (8.36) is called the *master stability equation*. We then introduce the *master stability function* (MSF):

$$\mathcal{M}(\xi) \quad (8.37)$$

which associates the Maximum Liapunov Exponent (MLE) of (8.36) to any value of ξ . Maximum Liapunov Exponents (Ott, 2002) are well-defined measures of the asymptotic growth rate of a perturbation about a trajectory (in this case $x_s(t)$). They are negative when the perturbation asymptotically decays back to the trajectory and positive when it

asymptotically departs from the trajectory. They should not be confused with eigenvalues of the Jacobian evaluated at the solution. We then have:

Theorem 8.8 *The totally synchronous solution is stable if*

$$\mathcal{M}(-S\lambda_k) < 0 \quad (8.38)$$

for all transverse eigenvalues λ_k , $k = 2, \dots, N$.

This condition must be satisfied for all transverse eigenvalues, not just one.

It is important to emphasize that (8.36) has dimension n , as well as (8.31). Therefore, with respect to the original problem (8.33) we have achieved a dimensional reduction from $N \times n$ to $2 \times n$, where the number 2 comes from counting both the dynamics of the synchronous evolution (8.31) as well as the dynamics of the decoupled modes (8.35). This is especially significant when the number of network nodes N is very large.

There is also another important implication of the parametrization (8.36). Namely, for a given choice of the functions F and H , we can compute *a priori* the region of the complex plane Ω within which $\mathcal{M}(\xi) < 0$, and then test whether the transverse eigenvalues of a given Laplacian matrix L belong to Ω . Accordingly, the condition for stability of the synchronous solution becomes

$$-S\lambda_k \in \Omega, \quad k = 2, \dots, N. \quad (8.39)$$

This has the significant advantage that computing Ω needs to be done only once. For a known Ω , assessing stability of the complete synchronous solution for a given network topology simply requires checking whether the transverse eigenvalues of the Laplacian multiplied by $-S$ fall into the region Ω . This gives a quick stability check for many different network topologies at once.

As a useful rule of thumb, it is suggested that when Ω is bounded, the more widely the transverse eigenvalues are spread, the less likely it is for all of them to lie in Ω , so stability is less likely. One plausible measure of the spread is the standard deviation of the eigenvalues. Barahona and Pecora (2002); Nishikawa and Motter (2010) propose this as a synchronizability index of a given network topology.

The main limitations of the master stability function approach described in this section are the assumptions of identical node dynamics and generalized diffusive coupling in (8.29) leading to the Laplacian dynamics which is not widespread in biology, as well as the impossibility of applying this analysis to cluster synchronization, which is of fundamental importance for biological systems.

8.11 A master stability function approach for cluster synchronization

A large body of work by Pecora, Sorrentino and others, e.g. (Sorrentino and Ott, 2007; Sorrentino et al., 2016; Siddique et al., 2018; Blaha et al., 2019; Della Rossa et al., 2020;

Panahi et al., 2021; Lodi et al., 2021, 2024), has focused on extending the master stability function approach to cluster synchronization. The first paper to accomplish this goal for an arbitrary network topology was (Pecora et al., 2014), which, however, focused on the case of symmetries produced by the graph automorphism group, and on orbital partitions. Here we are mostly interested in equitable partitions arising from the more general fibration symmetries, a case considered in (Panahi et al., 2021; Lodi et al., 2021). The general set of dynamical equations considered is:

$$\dot{x}_i = F(x_i) + \sum_{j \neq i} A_{ij} H(x_j), \quad i = 1, \dots, N. \quad (8.40)$$

These equations are not in the Laplacian form of (8.28). The case when the adjacency matrix is symmetric is studied in (Panahi et al., 2021) and the general case is studied in (Lodi et al., 2021). The method presents similar advantages to those described above, namely:

- (i) It can be applied to study stability of either periodic or chaotic cluster synchronous solutions.
- (ii) It can be applied to any given network and to any equitable partition (minimal or not).
- (iii) It leads to a dimensional reduction of the stability problem into subproblems of smaller dimension.

However, it is important to stress that, unlike (Pecora and Carroll, 1998), the extent of the dimension reduction varies from network to network, and can be limited.

For a given adjacency matrix A we partition the set of network nodes V into $C \geq 1$ equitable clusters, C_1, C_2, \dots, C_C , so that $\cup_{k=1}^C C_k = V$ and $C_k \cap C_\ell = \emptyset$ for $k \neq \ell$, with the condition of balanced coloring:

$$\sum_{h \in C_\ell} A_{ih} = \sum_{h \in C_\ell} A_{jh}, \quad \forall i, j \in C_k, \quad \forall C_k, C_\ell \subset V. \quad (8.41)$$

We let $|C_k| = n_k$ be the number of nodes in cluster $k = 1, \dots, C$, so that $\sum_{k=1}^C n_k = N$. Given such an equitable partition we can assume, without loss of generality, that the network nodes are ordered so that the first n_1 nodes are those in cluster C_1 , followed by the n_2 nodes in cluster C_2 , and so on. Then we construct *indicator matrices* E_k :

$$E_1 = \begin{pmatrix} I_{n_1} & 0 \\ 0 & 0_{N-n_1} \end{pmatrix} \quad E_2 = \begin{pmatrix} 0_{n_1} & 0 & 0 \\ 0 & I_{n_2} & 0 \\ 0 & 0 & 0_{N-(n_1+n_2)} \end{pmatrix} \quad \dots \quad E_C = \begin{pmatrix} 0_{N-n_C} & 0 \\ 0 & I_{n_C} \end{pmatrix} \quad (8.42)$$

where I_{n_i} is the identity matrix of size n_i , and 0_{n_i} is the zero matrix of size n_i .

We have seen that an equitable partition defines an invariant subspace for (8.40), the synchrony subspace of the cluster (also called the cluster synchronization manifold). The dynamics on this subspace gives the time evolution of the clusters (Golubitsky et al., 2005b) $\{s_1(t), s_2(t), \dots, s_C(t)\}$, where $s_1(t)$ is the synchronous solution for all nodes in cluster C_1 , $s_2(t)$ is the synchronous solution for nodes in cluster C_2 , and so on.

The $C \times C$ quotient matrix Q is defined so that, for each pair of equitable clusters C_k and

C_l ,

$$Q_{kl} = \sum_{j \in C_l} A_{ij} \quad i \in C_k. \quad (8.43)$$

The quotient matrix describes a quotient network, in which all nodes in each equitable cluster collapse to a single quotient node. Assuming that (8.40) evolves on the synchrony subspace, and averaging over all the nodes in each cluster, we can derive the equations for the time evolution of the quotient network:

$$\dot{s}_k(t) = F(s_k(t)) + S \sum_{l=1}^C Q_{kl} H(s_l(t)), \quad k, l = 1, 2, \dots, C \quad (8.44)$$

where the n -dimensional vector $s_k(t)$ represents the state of the quotient network node $k = 1, \dots, C$.

To investigate the stability of the cluster synchronous solution, we consider a small perturbation $\delta x_i = (x_i - s_k)$, $i \in C_k$. Linearizing (8.40) about (8.44), we can write

$$\delta \dot{x}(t) = \left[\sum_{c=1}^C E_c \otimes DF(s_c(t)) + S \sum_{c=1}^C AE_c \otimes DH(s_c(t)) \right] \delta x(t) \quad (8.45)$$

in the nN -dimensional vector $\delta x(t) = [\delta x_1^T(t), \delta x_2^T(t), \dots, \delta x_N^T(t)]^T$ and the indicator matrices defined in (8.42).

We are interested in the possibility that the stability problem for the Nn -dimensional system (8.45) can be decoupled into a set of lower-dimensional equations. Equation (8.45) is analogous to (8.33), introduced in the previous section for the case of complete synchronization. However, decoupling (8.45) into a set of lower dimensional equations is typically a much more complex task. The challenge is to find a transformation matrix T that transforms (8.45) into a set of lower dimensional equations of lowest dimension. (In the case of (8.33) this matrix T was easily constructed as the matrix of the eigenvectors of the Laplacian matrix L .)

Panahi et al. (2021) and Lodi et al. (2021) differ for the specific technique used to construct the matrix T , where the approach of (Panahi et al., 2021) is faster but works only when the matrix A is symmetric, while the approach of (Lodi et al., 2021) is slower but can be applied to the general case. Details of the construction of either transformation matrix T are in (Panahi et al., 2021) and (Lodi et al., 2021). Here we take the approach of (Panahi et al., 2021) and directly present the final result: the linearized system (8.45) is transformed into

$$\dot{\eta}(t) = \left[\sum_{k=1}^C J_k \otimes DF(s_k(t)) + \sum_{k=1}^C BJ_k \otimes DH(s_k(t)) \right] \boldsymbol{\eta}(t). \quad (8.46)$$

Here B is the transform of A , that is, $B = T^{-1}AT$, and the matrices J_k are the transforms of E_k , so $J_k = T^{-1}E_kT$, where $k = 1, \dots, C$. The matrix B is block-diagonal in $r \geq 2$ blocks and the matrices J_k are diagonal, so the dimensional reduction is determined by the structure of the r blocks into which B is decomposed. Thus (8.46) decouples into $r \geq 2$ independent equations of smaller dimension, in the r independent blocks of the block-diagonal matrices J_k and BJ_k . The structure of these blocks conveys important information

about key properties of the stability problem, and in particular about the possible breakings of the clusters that may lead to the emergence of other synchronization patterns.

As already stated, there is always one C -dimensional symmetry preserving block, which we label $k = 1$; this is associated with dynamics parallel to the synchrony subspace. The remaining $k = 2, \dots, r$ symmetry-breaking blocks are associated with dynamics transverse to the synchrony subspace. This is analogous to what we saw earlier in this chapter for equilibria, the difference being that for equilibria we had symmetry preserving and symmetry breaking eigenvalues and eigenvectors, and in the case of synchronous oscillations we have symmetry preserving and symmetry breaking modes. The condition for stability of the cluster synchronous solution is that all the maximum Liapunov exponents associated with the symmetry breaking blocks $k = 2, \dots, r$ are negative.

8.11.1 Examples of cluster stability analysis

Next, we present two examples to illustrate how this procedure works. First, we consider the $N = 4$ -dimensional network with $C = 2$ clusters shown in Fig. 8.16(a). The nodes are colored according to the cluster to which they belong. The synchrony subspace is $\Delta = \{x_1, x_1, x_2, x_2\} : x_1, x_2 \in \mathbb{R}^n$.

Figure 8.16 shows that applying the transformation T from (Panahi et al., 2021) to the matrices A , E_1 , and E_2 reduces the $4n$ -dimensional stability problem of (8.45) into $r = 2$ separate blocks: a $2n$ -dimensional equation corresponding to the quotient dynamics (symmetry preserving block) and a $2n$ -dimensional equation corresponding to the transverse dynamics (symmetry breaking block). Only the latter is responsible for stability of the cluster synchronous solution, i.e., it is the maximum Liapunov exponent associated with this latter block that determines whether the cluster synchronous pattern in (a) is stable. Figure 8.16b represents the adjacency matrix A and the indicator matrices E_1 and E_2 , where a black dot corresponds to an entry equal to 1 and the absence of a dot corresponds to a zero entry. Figure 8.16c represents the two decoupled ‘networks’ obtained after applying the transformation, and this is consistent with the block structure of the matrices B , J_1 , and J_2 in (d).

We may wonder why there is only one symmetry breaking block and not two. In fact, since the equitable partition in (a) has $C = 2$ clusters, we might expect that stability of each of the two clusters corresponds to a separate symmetry breaking condition. The reason why only one symmetry breaking condition occurs is that the two clusters are ‘intertwined’ (Pecora et al., 2014). That is, it is not possible to break the yellow cluster without also breaking the green cluster and *vice versa*. Thus either the two clusters are simultaneously synchronized, or they are simultaneously desynchronized. As a result, stability of the cluster pattern (a) depends only on one maximum Liapunov exponent, which corresponds to simultaneous breaking of the yellow and green clusters. There is no other way in which the cluster synchronous solution can be broken.

We now apply our approach to a network for which the clusters of the minimal equitable partition and the clusters of the minimal orbital partition do not coincide. This has $N = 8$ nodes, and is taken from (Kudose, 2009); we call it the *papillon network*. Figure 8.17 presents the case of the network minimal equitable partition with $C = 2$ clusters, and Fig.

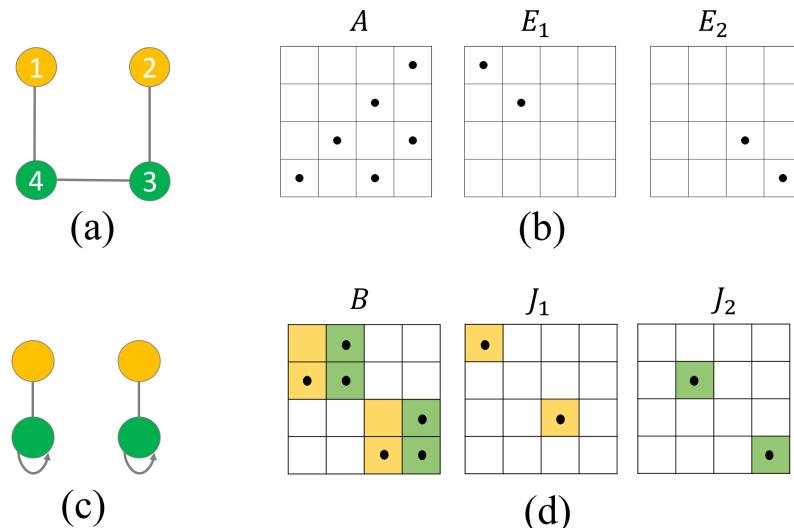


Fig. 8.16

Stability of the cluster synchronous solution. (a) An $N = 4$ node network with $C = 2$ clusters. Nodes are color-coded according to the clusters to which they belong. (b) The adjacency matrix of the network (A) and the two cluster indicator matrices ((E_1, E_2)) are shown graphically; a nonzero entry is indicated by a black dot. Panel (c) shows the quotient and transverse subnetworks after applying T . Subnetwork nodes are colored according to the cluster with which they are associated. (d) The set of matrices $\{B, J_1, J_2\}$ is obtained using the transformation T . The background color of each matrix entry indicates the cluster about which the dynamics is linearized.

8.18 presents the case of the network minimal orbital partition with $C = 3$ clusters. Both figures are similar in content and style to Fig. 8.16.

Figure 8.17 shows that applying the transformation T leads to a total of 5 blocks. The first block is 2-dimensional and corresponds to the quotient network (symmetry preserving block). The remaining four blocks, with dimensions 3, 1, 1, 1, correspond to the transverse dynamics (symmetry breaking blocks). The quotient block is the only 2-dimensional block. Stability of the cluster synchronous solution depends on the maximum Liapunov exponents associated to the four symmetry breaking blocks. Each of these symmetry breaking blocks has a nice geometric interpretation in terms of possible breakings of the cluster synchronous solution in (a), which may lead to the emergence of a different (non-minimal) equitable partition of the network nodes.

The 3-dimensional transverse block corresponds to a left/right symmetry breaking in which the cluster pattern in (a) is broken along the vertical symmetry line that divides the network into two halves of four nodes each. This may happen without breaking synchronization of each of the two pairs of red nodes on the left and on the right of the papillon in (a). In addition, there are three other symmetry breakings of the red cluster alone that may disrupt the cluster synchronization pattern in (a). One is for the two center red nodes to

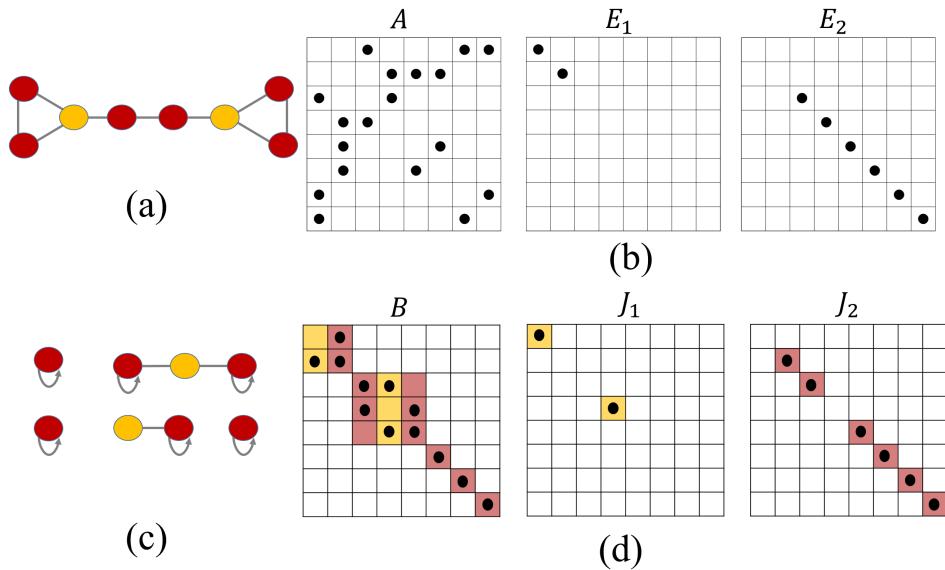


Fig. 8.17

The papillon network with minimal balanced coloring/equitable partition. (a) An $N = 8$ -dimensional network with $C = 2$ equitable clusters. Nodes are color-coded according to the clusters to which they belong. (b) The adjacency matrix of the network (A) and the two cluster indicator matrices E_1, E_2 are shown graphically, in which a nonzero entry is indicated by a black dot. (c) shows the quotient and transverse subnetworks after applying the transformation T . Subnetwork nodes are colored based on the cluster to which they are associated. (d) The set of matrices $\{B, J_1, J_2\}$ obtained by using the transformation T . The background color of each matrix entry indicates the cluster about which the dynamics is linearized.

depart from the four outer red nodes (the corresponding new cluster synchronization pattern is shown in panel (a) of Fig. 8.18). The remaining two correspond to the two possible ways in which the four outer red nodes can be broken into two halves of two nodes each, either along the top-bottom horizontal symmetry line or along the top left-bottom right diagonal symmetry line. In order for the cluster synchronous pattern in (a) to be stable, all the four maximum Liapunov exponents corresponding to these symmetry-breaking blocks should be negative, thus preventing any of these possible breakings from happening.

Figure 8.18 shows the same analysis applied to the papillon network but for the minimal orbital partition shown in panel (a). In this case, applying the transformation T leads to a total of 4 blocks: one 3-dimensional block corresponding to the quotient network (symmetry preserving block), and three blocks with dimensions 3, 1, 1 corresponding to the transverse dynamics (symmetry breaking blocks). Stability of the cluster synchronous solution depends only on the maximum Liapunov exponents associated to the three symmetry breaking blocks, each of which has a nice geometric interpretation in terms of possible breakings of the cluster synchronous pattern (a).

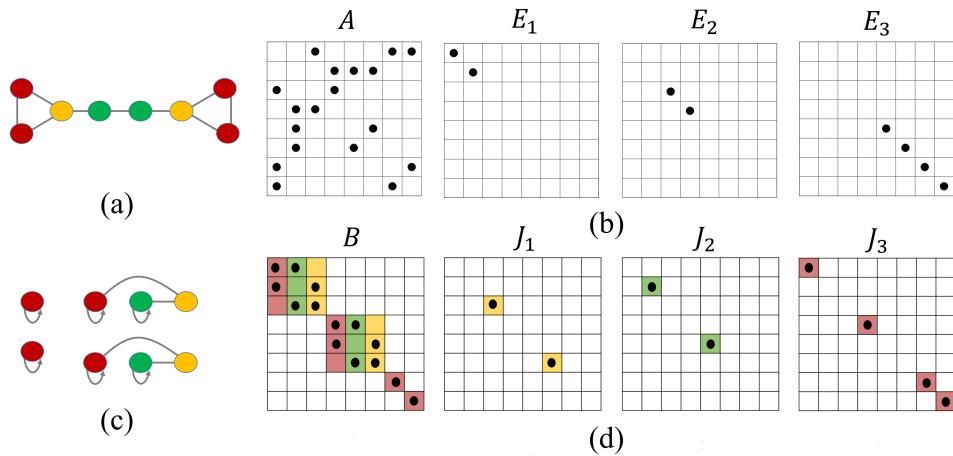


Fig. 8.18

The papillon network with minimal orbital partition. (a) An $N = 8$ -dimensional network with $C = 3$ orbital clusters. Nodes are color-coded according to the clusters to which they belong. (b) The adjacency matrix of the network (A) and the three cluster indicator matrices E_1, E_2, E_3 are shown graphically; a nonzero entry is indicated by a black dot. (c) The quotient and transverse subnetworks after applying the transformation T . Subnetwork nodes are colored based on the cluster to which they are associated. (d) The set of matrices $\{B, J_1, J_2, J_3\}$ obtained by using the transformations T and \tilde{T} , respectively. The background color of each matrix entry indicates the cluster about which the dynamics is linearized.

Similarly to Fig. 8.17, the 3-dimensional transverse block corresponds to a left/ right symmetry breaking in which the cluster pattern in (a) is broken along the vertical symmetry line that divides the network into two halves of four nodes each. This may happen without breaking synchronization of the two pairs of red nodes on the left and on the right of the papillon in (a). In addition, there are other two symmetry breakings of the red cluster that may disrupt the cluster synchronization pattern in (a). These correspond to the two possible ways in which the four outer red nodes can be broken into two halves of two nodes each, without breaking the green and yellow clusters: either along the top-bottom horizontal symmetry line or along the top-left-bottom-right diagonal symmetry line.

In order for the cluster synchronous pattern in (a) to be stable, all the three maximum Lyapunov exponents corresponding to these symmetry-breaking blocks should be negative. The main difference from Fig. 8.17 is that the 6-node red cluster in Fig. 8.17 already appears to be broken into two clusters in Fig. 8.18a. This corresponds to an increase in the dimension of the quotient network (which has now $C = 3$ nodes) and a corresponding decrease in the dimension of the transverse blocks. Hence, there are situations for which the cluster synchronization pattern in 8.18a is stable but the cluster synchronization pattern in 8.18b is not, because this requires stability of one additional maximum Lyapunov exponent.

8.12 Structural stability: changes to equations, parameters, and graphs

‘Stability’ is a word with numerous different meanings. Several other important properties of network models have an air of stability as well as robustness about them, and it is important not to confuse them with stability as discussed so far in this chapter. All of them are important to biological modeling, so for completeness we mention them here and make a few pertinent remarks. They include:

(1) **Synchronizability:** *Stability against small changes to initial conditions.* This is the stability concept described in previous sections of this chapter, and it is perhaps the most basic. A solution of an ODE is stable in this sense if a small change in initial conditions leads to a solution that is the same as, or at least ‘very close to’ the original solution. It is called synchronizability when the state is synchronous. This is important for biology because the environment is constantly changing, and such changes should not cause radical differences in the behavior predicted by a model.

(2) **Hard structural robustness:** *Stability against (small) changes in the model equations,* when these remain consistent with the same underlying graph. This is a form of *structural stability* in the sense of Smale (1963), tailored to the graph setting. There it is known as *robustness*, see Definition 5.9. An apparently weaker condition, which turns out to be equivalent for equilibria (and probably for more complex dynamical states) is *rigidity*. This notion of stability is independent of (1). An unstable state can nonetheless be structurally stable if it persists after small perturbations to the model ODE. It will still exist but remain unstable. This notion is important for biology because model equations are seldom known exactly; they are usually phenomenological models involving functions and parameters that have been chosen to offer useful insights but do not reflect reality with high precision. This form of stability preserves the underlying graph representing known biological interactions but allows minor changes to the functions in the model.

(3) **Soft structural robustness:** *Stability against changes to parameters.* This is a special form of (2): the allowed perturbations to the model are already built in as numerical parameters, which may not be known accurately. Changing these parameters changes the equations, which is why (2) applies, but the range of changes considered is more restrictive. The relevant mathematical concepts are hyperbolicity (Guckenheimer and Holmes, 1983), which ensures that in typical circumstances, sufficiently small changes to a model do not destroy equilibria or periodic states (even if these are unstable in the sense of (1) above). This stability notion is important for biology because biology is disordered and random; for example, the binding of molecules depends on many things, and it cannot be assumed to be uniform in biological systems. See Chapter 21.

(4) **Hard structural robustness:** *Stability against graph modifications.* This is an interesting but relatively new area of mathematics, which becomes relevant when we are uncertain whether the graph being assumed is an accurate representation of all relevant biological interactions. Even when this is the case, biological events such as evolution and mutations may change the graph. Again, hyperbolicity can indicate this kind of stability as

long as the graph modifications lead to small changes in the ODE. Simulations are often the only way to study this type of question.

In this book, we reserve the word ‘stable’ for (1), unless otherwise indicated.

Extending Fibrations Beyond Graphs to Hypergraphs

Up to this point, we have discussed systems that can be described by directed (multi)graphs with two-body interactions associated with graph edges. Modeling biological networks often requires a more specific type of structure that prescribes the many-body interactions among the biological units. These kinds of interactions are widespread in biology and are not described by graphs when (as is often assumed) each edge corresponds to a specific function of a single input variable. A popular way to impose such constraints is to use a hypergraph. Hypergraph equations are admissible but satisfy extra conditions on the terms that appear in the ODE. In this chapter, we describe how the techniques we have previously discussed can be applied, *mutatis mutandis*, in this more general setting. We also discuss other generalizations of fibrations to weighted, multiplex, and multilayer networks.

9.1 Introduction

So far, we have discussed many different types of systems that can all be described in the form of directed graphs (networks): a set of entities (protein, genes, neurons, cells, chemical compounds, etc.) with directed edges between them representing some form of binary action (control, relation, enhancement, activation, repression, excitation, inhibition, etc.). The notions of symmetries that we have taken into consideration mainly target this simple (yet rich) kind of structure. We should, however, notice that in many cases (most probably all) biological networks are more complex, and describing them merely as graphs would be an oversimplification. In this chapter, we want to take some of these complex structures into consideration; the most relevant to biology are hypergraphs and weighted networks. We discuss in each case whether and how the techniques described in the book can still be applied to them. These applications remain open for additional theoretical developments.

9.2 Hypergraphs

While graphs serve as a powerful tool for modeling complex systems, they are not always the best choice, because in many cases interactions between components are not binary but can involve more than two components at the same time. As discussed in Section 3.2, a general admissible ODE for an ordinary (multi)graph is not limited to binary interactions:

if a node has three inputs, for instance, then *all possible functions* of the three tail nodes can occur in an admissible ODE. If the input set has m edges, then m -ary interactions are, in principle, permitted; $m + 1$ if we count the node variable.

The problem is that in many areas of application, standard models involve specific functional forms for interactions. The resulting admissible ODEs include all possible admissible equations, many of which do not have the required specific form used by modelers in the area concerned. While the theory of admissible equations is important from the mathematical point of view, leading to general theorems about admissible dynamics that apply to *all* models (see Section 3.4 and discussion in Section 5.9), a modeler in a specific discipline would be more interested in a representation of the ODE that describes the dynamics of interest in a particular practical application.

Hypergraphs (Berge, 1973) are a popular solution to this problem and are also adopted in data modeling of higher-order interactions (Battiston et al., 2020; Millán et al., 2025). Hypergraphs provide a systematic way to represent specific *restrictions* on the form of model ODEs, which are not consequences of the usual network structure.

For instance, in social network analysis, a hypergraph can represent a group of people (vertices) participating in an event (hyperedge), a situation that cannot be accurately represented by a simple graph. However, hypergraphs also find a very natural use as models for biological networks, such as protein-protein interaction networks, gene regulatory networks, and metabolic networks. In all of these networks, complex interactions often involve more than two entities, making hypergraphs a suitable modeling tool.

A biological example is shown in Fig. 9.1, taken from the KEGG Orthology database (Kanehisa et al., 2004); it represents the metabolic pathway of fructose and mannose metabolism. A metabolic pathway like this is a complex network connecting metabolic reactions, genes, and compounds. Reactions typically have many inputs (reactants) and/or outputs (products). A general admissible ODE for this graph would permit numerous models (infinitely many) that do not adequately capture the biological meaning. Some of these models are realistic representations, but some are not. The modeler is more interested on a particular representation with particular types of interactions.

Generally speaking, hypergraphs are generalizations of graphs where each edge (called a ‘hyperedge’ or ‘hyperarc’) involves an arbitrary number of nodes instead of just two. This idea can take a number of flavors, depending on the actual situation we want to model. For instance, Fig. 9.2 is a standard depiction of an *undirected hypergraph*: hyperedges here are just sets of nodes; while some edges contain exactly two nodes (like h_2 in the picture), which is what happens in a standard undirected hypergraph, there are also hyperedges containing more (h_1 and h_3 , in the example) or less (h_2) nodes.

Let us focus on another type of hypergraph that is particularly common in the biological context: in biological systems, hypergraphs are typically directed: a *directed hypergraph* has hyperedges identified with pairs (S, T) where S and T are nonempty sets of nodes: the set S represents the source (also called ‘domain’ or ‘tail’) of the hyperedge, and the set T represents its target (also called ‘codomain’ or ‘head’). Figure 9.3 (left) shows an example of a directed hypergraph: in this example, all hyperedges have exactly one target.

The easiest way to approach the problem of extending the notion of fibration to hyper-

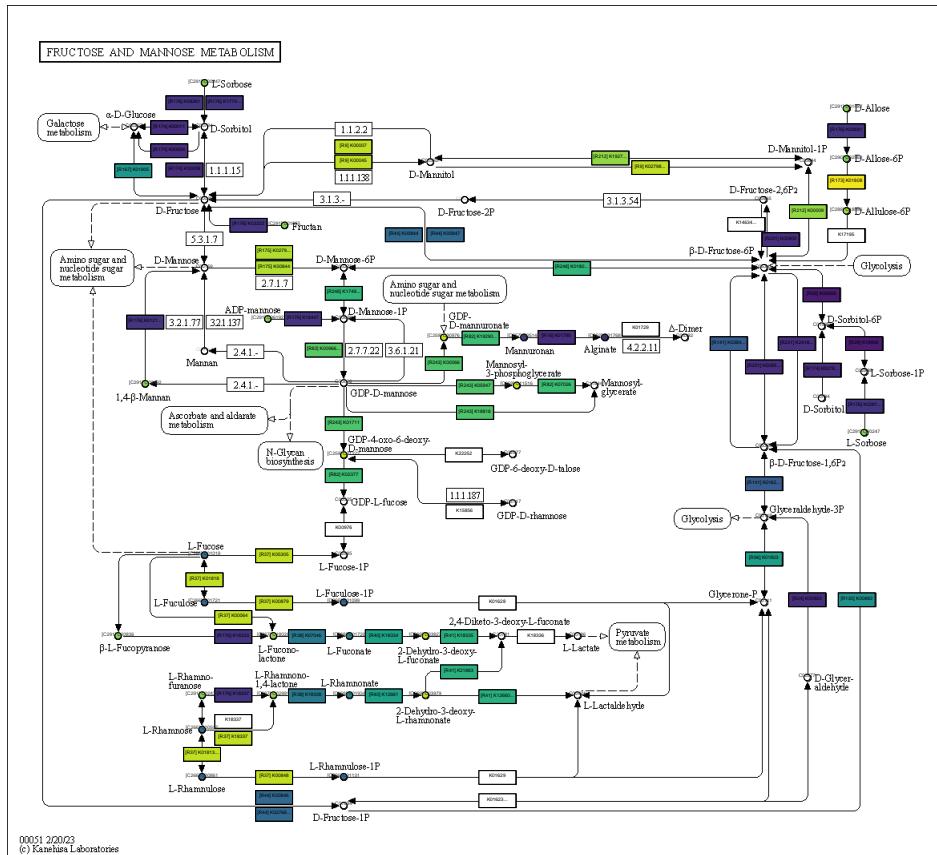
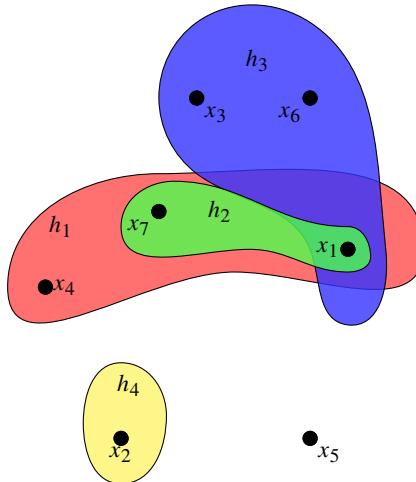


Fig. 9.1

Example of a complex hypergraph in biology. The metabolic pathway describing fructose and mannose metabolism from KEGG. Figure reproduced from (Kanehisa et al., 2004). Copyright © 2004, Oxford University Press.

graphs, as also suggested in (Preti et al., 2024), is to represent hypergraphs in the form of a *factor graph*, a bipartite directed graph: we turn every node and hyperedge of the hypergraph H into a node of the factor graph $[H]$, and use edges in $[H]$ to represent sources and targets. Figure 9.3 (right) shows the factor graph representation $[H]$ of the hypergraph H appearing above in the same figure: the seven nodes of H (v_1, \dots, v_7) are also nodes of the factor graph $[H]$, but in the factor graph we also have one extra node for every hyperedge. These extra nodes are called factors and are represented as rectangles (to highlight the fact that the factor graph is bipartite). Each factor is linked to its sources and targets by incoming and outgoing edges in the factor graph: for instance, hyperedge h_1 has source $\{v_1, v_2, v_4\}$ and target $\{v_3\}$. In the ODE, the factor represents the input function of the factor's output node.

Now, besides being convenient, it can be formally shown (Boldi, 2023) that graph

**Fig. 9.2**

Example of undirected hypergraph. It contains seven nodes and four hyperedges.

fibrations from $[H]$ are equivalent to hypergraph fibrations from H , provided that typing is taken into account. Typing here is needed because we do not want to send nodes to factors or factors to nodes, something that we may end up doing without typing: recall that both factors and nodes of H are nodes of $[H]$, so a homomorphism that does not consider types may freely mix factors and nodes, a behavior that we need to avoid.

Hence the blueprint for hypergraph fibration symmetries would be to use the map $[-]$ (technically, an equivalence functor between the category of hypergraphs and the category of typed graphs) to get back to the world of typed graphs, and then use the machinery we have already set up to discuss symmetries in that context.

A study of synchrony patterns in hypergraphs with identical nodes, which also considers their stability, can be found in (Aguilar et al., 2023).

9.3 Graph, hypergraph, and admissible graph representations of ODEs

The representation of the ODE system is crucial to the application of fibrations. Here, we review the three representations discussed so far and explain when they are useful and when they are not.

The purpose of representing an ODE as a graph is to enable the calculation of balanced colorings. These balanced colorings must align with the cluster synchrony present in the ODE. This alignment is essential for the practical application of fibrations in typical scenarios. However, there is more to discuss beyond this initial explanation.

As remarked before, each area of application has its own rules for turning graphs (or multigraphs or hypergraphs) into model equations. These rules may differ from those

used in this book, and how the graph is drawn can differ from one area to another. We further illustrate this issue by discussing the graph, hypergraph and admissible graph representations using a genetic and metabolic network, an important hypergraph to which we return in Chapter 16.

9.3.1 Graph representation

To review the assumption of the graph representation, we return to the UxuR-LgoR fiber in *E. coli* exemplified in Fig. 2.1 and whose ODE is represented in (2.1), as an example of a system with two-body interactions. There are some hidden assumptions that allow the graph to be a valid representation of the ODE and vice versa (see also Chapter 21). If couplings are additive and two-body, for example—as we have often assumed in this book—then each edge in the graph determines only the contribution of one input to the input function of the node.

Each edge-type in the graph Fig. 2.1 represents either an activator of the form $d \frac{x^2}{(1+x^2)}$ or a repressor $b \frac{1}{(1+x^2)}$. Additionally, in the ODE (2.1) we use the same constants b and d in x_2 and x_3 interaction terms, leading to the same link in the graph. That is, $e_1 = e_3$ and $e_4 = e_5$. This consistency is necessary to generate the fibration; if the parameters were different, the fiber would be gone (effectively ‘bye-bye’ to the fiber due to the symmetry breaking discussed in Chapter 21). This may seem disappointing, as we dedicate an entire chapter to explore this aspect. Furthermore, the internal degradation dynamics are assumed to be the same for nodes 2 and 3 since we use the same constant c in the ODE. The equal internal dynamics is represented in the graph by the same symbol for nodes 2 and 3.

There is an extra assumption for the graph to be a valid representation of the ODE: we need to specify how different input links are combined in a node’s input function. In the assumption presented in (2.1), the combination is additive; however, any other form of input function can also be valid. The key point of the graph representation is that the type of input function must remain the same for every node. So, all the links are combined in the same way for every node’s input function. This condition includes two-body interactions but also many-body interactions. That is, a many-body system with the same type of input function for all nodes still admits a graph representation. Otherwise, a factor graph (hypergraph) is needed to specify the different input functions (even in the pair-wise case).

Thus, when more specific functions are required, the graph representation must be augmented. In such scenarios, a hypergraph and its corresponding factor graph would be appropriate to represent the meaning of the different input functions, even for two-body interactions. Consequently, the entire book should be rewritten to focus on hypergraphs. Fortunately, many of the results can be adapted from graphs to hypergraphs, *mutatis mutandis*, although hypergraph fibrations remain an exciting and open area of research.

On the contrary, the more general theory of admissible ODEs from Section 3.4 is not to assign each term in a formula to a corresponding arrow. Instead, the input isomorphism class of the input set determines which components of the admissible ODE use the same function (with corresponding variables assigned according to the relevant input isomorphisms) but

otherwise, this function is not specified. Further discussed in this section, this representation is the *admissible graph*.

The extra structure imposed by passing from an admissible graph to a hypergraph should not be viewed as *extending* the range of admissible models, but as *restricting* it to a smaller class of models whose form corresponds more closely to the biology and the particular type of ODE describing the system.

9.3.2 Hypergraph representation

As a typical example, we consider two metabolites m_1 and m_2 in a bidirectional biochemical reaction metabolized by enzymes E_1 and E_2 , which are, in turn, repressed, as in the ODE:

$$\begin{aligned}\dot{m}_1 &= -E_1 m_1 + E_2 m_2 \\ \dot{m}_2 &= E_1 m_1 - E_2 m_2 \\ \dot{E}_1 &= -\alpha E_1 + \frac{1}{1+E_1} \\ \dot{E}_2 &= -\alpha E_2 + \frac{1}{1+E_1}\end{aligned}\tag{9.1}$$

This ODE can be represented by a standard biochemical representation as in Fig. 9.4a; by a hypergraph, Fig. 9.4b; or by a network following the conventions of the admissible graph framework in Section 3.4 from (Golubitsky and Stewart, 2023)), Fig. 9.4c. The hypergraph in (b) conveys the information to reproduce exactly the form of the ODE, including the detailed functions involved in the ODE. In the figures, the node variables m_1, m_2, E_1, E_2 are written inside their node symbols.

We discuss now how these different graphical representations relate to the ODE (9.1). In the biochemical representation Fig. 9.4a, the variables E_1, E_2 are replaced by $v_1 = m_1 E_1$ and $v_2 = m_2 E_2$. Both inputs and outputs contribute to the terms of the ODE. The mass balance condition implies that an outgoing arrow contributes a negative term. For example node m_1 has an output to v_1 representing $-m_1 E_1$, and an input from v_2 representing $+m_2 E_2$. The rules for converting the graph to the ODE are standard in the area of metabolic networks. However, this is neither a graph representation nor a hypergraph. It is just a convenient representation of the biochemical pathways. Thus, it cannot be used in a fibration analysis to calculate the balanced coloring. It does not represent the ODE.

In the hypergraph representation of Fig. 9.4b, separate terms in the ODE, and the variables they combine, are drawn as smaller squares that contain $-$, $+$. These represent the factors, which are the input functions of the output node of the factor, as shown on the right. Their source nodes are combined using the appropriate function, and the output is sent to the target of the output arrow. For example, the term $-E_1 m_1$ in the \dot{m}_1 component of the ODE is represented by the two arrows targeting the top left-hand square marked ' $-$ ', and the output arrow to node m_1 indicates that this term appears in the \dot{m}_1 component. It is assumed, by convention, that these terms are added together in the model ODE.

The ODE (9.1) does not have \mathbb{Z}_2 symmetry, because the same term $\frac{1}{1+E_1}$ occurs in the last two components. The situation is similar to that of the metabolator and Smolen circuits, see Section 8.8.3, and we seek a fibration rather than an automorphism. Inspecting for input isomorphisms, we see that there is a fibration in which nodes m_1 and m_2 are synchronized

and nodes E_1 and E_2 are synchronized. To support this claim, we show the input trees (to level 2) of nodes 1 and 2 in Fig. 9.5. The input trees of E_1 and E_2 are standard AR loops.

This is the correct representation of the ODE that yields a balanced coloring in agreement with the synchronous dynamics. In practical applications, either a graph (provided the aforementioned conditions are met) or a hypergraph should be utilized. We then ask: what is the purpose of the admissible graph representation (Section 3.4) in Fig. 9.4c?

9.3.3 Admissible graph representation

We recall that the admissible graph is constructed as a graph of 'influence' between the nodes. Thus, an edge from i to j appears in the admissible graph when there is an interaction term in the ODE of j where i appears.

In this book we normally use a multigraph to represent admissible graphs—but, as previously explained, its edges do not represent single-body interactions, as is the case in the graph representation. Instead, our conventions on admissible ODEs, which focus on input isomorphisms, effectively treat the multigraph as a special kind of hypergraph. The entire input set of arrows (together with their source nodes) is treated as a single hyperedge representing the collective interactions created by the source nodes. No other hyperedges occur.

Unlike a hypergraph, there is one extra feature: if the input set includes more than one arrow of the same type, the function f_c is symmetric in the corresponding entries. The intuition is that the node concerned 'does not know' which nodes send which signal. It just processes the entire set of signals according to f_c . These constraints are formalized by 'vertex symmetries', which permute source node variables for arrows of the same type. An example occurs in equation (8.11) for the lock-on circuit in Section 8.8.1.

In the admissible graph representation Fig. 9.4c, the small square nodes representing functional combinations no longer appear. Input arrows from node j to node i show that \dot{x}_i depends on x_j . The node symbol itself shows that \dot{x}_i depends on x_i . Thus $\dot{x}_i = f_i(x_i, x_{j_1}, \dots, x_{j_k})$, where x_{j_1}, \dots, x_{j_k} run through the source nodes of the input arrows. Arrows drawn in the same style (solid, dashed, ...) indicate how these variables correspond in the component functions; however, this information applies *only* to input isomorphic nodes. The graph then encodes not a specific ODE, but the set of all admissible ODEs. In this case, the admissible ODEs have the form

$$\begin{aligned}\dot{m}_1 &= F(m_1, E_1, m_2, E_2) \\ \dot{m}_2 &= F(m_2, E_2, m_1, E_1) \\ \dot{E}_1 &= G(E_1, E_1) \\ \dot{E}_2 &= G(E_2, E_1)\end{aligned}\tag{9.2}$$

for arbitrary functions F, G .

Remark We can also require the function F to be symmetric under interchange of the first two components, and of the last two, because $m_1 E_1 = E_1 m_1$ and $m_2 E_2 = E_2 m_2$. But this feature is rather incidental, and we ignore it here.

In each component, the first variable is the node variable; for example in the \dot{m}_1 component

is the function $F(m_1, E_1, m_2, E_2)$. In (9.1), there is no explicit term involving m_1 on its own, so in an intuitive sense, the ‘internal dynamic’ is zero. But, as previously explained, the formalism of admissible graphs does not associate individual arrows with individual terms in the ODE.)

If in (9.2) we set

$$F(u, v, w, x) = -uv + wx \quad G(y, z) = -\alpha y + \frac{1}{1+z}$$

(where u, v, w, x, y, z are ‘dummy’ variables used to define the form of F and G), then we recover (9.1). However, infinitely many other choices of F and G are possible in general. Which choices make biological sense depends on the modeling assumptions. Crucially, this graph has a fibration, corresponding to the cluster synchrony in which $m_1 = m_2$ and $E_1 = E_2$. We deduce that this cluster pattern is consistent with the ODE for *any* choices of F and G . This is a much stronger statement than its consistency with (9.1). It implies that the synchrony state is robust in the sense of Section 5.9.

But the price we pay for a stronger statement is that the graph does not tell us how to write down a specific model ODE; just the class of all admissible ones. So (c) is more useful for some theoretical purposes, such as classifying possible cluster synchrony patterns, but less so for modeling specific biological systems, especially in a quantitative manner.

9.3.4 Which representation do we use?

The answer to this question is subjective and depends on the perspective of the practitioner. In most cases, we recommend using graph representations when the conditions for their validity are satisfied; otherwise, the more general hypergraph should be used. In practical situations, a practitioner—such as a bioinformatician or a physicist—may be interested in a specific system, like the regulatory network of prostate cancer. This system is described by particular ODEs but not by the entire set of admissible ones. Therefore, the theory of admissible representations would be inappropriate in this context. For any practical application, the practitioner should always use a hypergraph, which, as shown in the above cases, always gives the correct balanced coloring that will be reproduced in experiments.

The admissible graph representation is more useful for mathematicians attempting to prove general theorems. A good example of this is the case of robustness under general modifications of the admissible equations (Section 5.9). For a robust system (in this technical sense), we can claim that the synchronous pattern arises from a fibration, which is a fundamental theorem in the fibration framework. This is a very significant result: *fibration implies synchrony, and synchrony implies fibrations*. This means that every time we find synchrony in a system, we can use fibrations to describe it. This is the major theoretical underpinning for the applications in Chapters 23, 24, and 25, where we will use fibrations to infer brain networks from synchronization data. But the theorem relies on the assumption of robustness for all general admissible models. It is not valid for balanced colorings on the hypergraph that do not satisfy the robustness condition. However, such colorings may still exist experimentally, which is what matters in all applications.

Robustness is, of course, an important issue for biology too. So results ought not to

depend on the precise formula in the model. However, network practitioners do not always appreciate this. In their support, a hypergraph can reflect known interactions, restricting the type of model more than by just assuming a fully general one.

Overall, the issues discussed in this section are fascinating, as they explore the challenges of transitioning from ideal mathematical theories to the realities of the physics of life. The core argument of this book is that once this tension is resolved, biology will become more akin to physics, with theoretical predictions leading to experiments rather than the other way around.

9.4 Weighted networks

Another very common class of networks, not only in the context of biology, is that of weighted graphs: a *weighted graph* is a network where each edge has a *weight*, a real number that represents the strength of the influence between the two nodes concerned. While this situation is extremely common, and the necessity to model it does not require any further discussion, weights are difficult to deal with in a purely combinatorial way, and at base of our approach is a definition of symmetry which is exclusively combinatorial (i.e., discrete) in nature. The same limitation also holds for automorphisms. Similarly, fibrations do not lend themselves easily to the weighted world.

A possible, quite brutal, way to treat weights is by *thresholding*: we establish a threshold θ , and just throw away all edges with weights smaller than θ , whereas the remaining edges are retained and their weight is discarded. This solution is arbitrary, and the results we obtain depend on the choice of θ . Thus, studies concentrate on varying θ and choosing an appropriate threshold where connectivity of the network arises. This naturally leads to a percolation problem. Such a thresholding process is popular in the analysis of networks in biology, specially in functional brain networks from fMRI data as treated in Chapter 25.

An alternative, more elegant, solution is *discretization*: we forget about fine-grained differences between weights, and consider edges with significantly different weights as if they represent two different types of interaction. In other words, we turn weights into types, provided that the weights are first suitably coalesced (or ‘coarse-grained’). For a concrete example, consider the weighted network of Figure 9.6.

In Fig. 9.7, we show three different ways to discretize the weights by turning them into types. For readability, types are displayed as edge labels, and are integers, but they should not be interpreted as weights any more, but as labels for types.

The three discretizations in Figure 9.7 are obtained as follows:

- (a) The type of edge is the same (1) regardless of the weight, i.e., $\eta(a) = 1$.
- (b) We distinguish between weights smaller than 0.5 and weights larger than or equal to 0.5, i.e., $\eta(a) = \lfloor 1 + 2w(a) \rfloor$.
- (c) We use a more fine-grained discretization, letting $\eta(a) = \lfloor 1 + 10w(a) \rfloor$.

In practice, the coarsest equitable partition (a) does not take weights into account *at all*: the symmetries we find are simply due to the graph structure. For instance, blue nodes all

have one incoming edge from blue and one from yellow; whereas yellow all have just one incoming edge from blue; the green node is the only one with indegree three.

Now, when we consider (b), we see that blue nodes are still together (they all receive a 2-edge from blue and a 1-edge from yellow), but the three yellow nodes are now not together in the same cluster any more: while 2 and 7 both receive a 2-edge from blue, 1 receives a 1-edge from blue, so they get different inputs. Intuitively, the different weights in the connecting edges make the two recipients different.

The more fine-grained the discretization becomes, the finer is the equivalence relation we obtain at the end. In (c) almost all edge types are different, and the few that are not are irrelevant to induce a symmetry.

Of course, even if in our example we use a linear form of discretization (in fact, translating weights into types by using their magnitude and deciding where to put boundaries), other forms of discretizations are entirely possible (e.g., logarithmic) and can make sense in some contexts.

This proposal has a number of evident shortcomings:

- Discretization is arbitrary, and how we discretize weights influences strongly the result.
- Every form of discretization is by its very nature discontinuous; hence, for weights that are ‘close to the border’ we must somehow decide which type they belong to, and how we take this decision may lead to largely different outcomes.
- Closely related to the previous item is the fact that there can be no overlap between types, and that each edge has but one type.
- There is no inherent form of ‘arithmetic’ between weights: receiving two edges of weights w_1 and w_2 from nodes of the same class is totally different from receiving one single edge with weight $w_1 + w_2$ from a node of the same class. This happens because lifting is a discrete property, that must hold for every single edge, so a node with two incoming edges will never ever be matched with a node with just one incoming edge, independently of how weights are discretized.
- The previous objection can be avoided to some extent by redefining the notion of a fibration, but only if the admissible ODEs have a suitable additive structure such as linear coupling. This is treated in (Aguiar and Dias, 2018, 2020; Sequeira et al., 2021, 2022).

9.4.1 Balanced coloring for a weighted network

The discretization and thresholding techniques discussed in the previous section enable the transformation of a weighted network into a binary network, which can then be analyzed using fibrations. Alternatively, we can redefine balanced coloring directly for the weighted network without filtering. While this approach does not yield a fibration, it often suffices in many real-world applications where the primary goal is to identify the balanced coloring of the network. Therefore, generalizing balanced colorings from unweighted to weighted networks can be adequate for numerous applications.

We can reinterpret the condition of balanced coloring for an unweighted network $G = (V, E)$ in matrix form. Let the adjacency matrix be $A_{ki} = \{0, 1\}$, and for simplicity assume

that all edges have the same type. Then i and j are balanced in clusters $C \in \mathcal{C}$ if, for all $C \in \mathcal{C}$, all pairs of distinct nodes $i, j \in C$, and all $D \in \mathcal{C}$, we have:

$$i \sim j \iff \sum_{k \in D:ki \in E} A_{ki} = \sum_{r \in D:rj \in E} A_{rj}. \quad (9.3)$$

Here $i \sim j$ means that i and j are in the same cluster.

The weighted case requires new variables and a modification of constraints. The adjacency matrix is modified using w_{ij} to denote the weight of edge $ij \in E$, and the condition of balanced coloring becomes:

$$i \sim j \iff \sum_{k \in D:ki \in E} w_{ki} = \sum_{r \in D:rj \in E} w_{rj}. \quad (9.4)$$

We acknowledge that this is one possible solution, which assumes that the colors are balanced by the sum of the weights. This assumption is reasonable if the system of ODEs represented by the weighted graph is additive in the weights of various input terms within the input function that defines the dynamics of each node. However, this assumption may not hold true in all cases. Each application should take into account different rules to achieve a balance in the weights. We revisit the issue of weighted networks in Chapter 21.

9.5 Multiplex and multilayer graphs

More often than not, biological systems have a heterogeneous structure. This heterogeneity mainly happens in two different ways (Hammoud and Kramer, 2020; Della Rossa et al., 2020).

- Edges of the network may bear different meanings: we already presented examples of networks with activation and inhibition edges, but the number of ‘edge types’ is not limited to two. Networks with different types of edges are often referred to as *arc-heterogeneous* or *multiplex*. Since these networks present different types of interactions (edges) between the same set of actors (nodes), they are conveniently represented by replicating the same set of nodes over different layers and assigning to each layer all edges of a given type. Figure 9.8 (left) shows an example of a network of five proteins with three different layers. The example in this figure is undirected, but directed graphs, or networks mixing directed and undirected layers, can also occur.
- Nodes of the network can also have different types; in this case the network is often said to be *node-heterogeneous* or *multi-layered*. For instance, in a metabolic network some nodes may represent genes, while others represent proteins or metabolites. Figure 9.8 (right) shows an example, for which each layer now contains all nodes of the same type. An important difference is that the nodes in different layers of the multilayer network of Fig. 9.8 (right) are different objects (genes, proteins, metabolites), while the nodes in different layers of the multiplex network in Fig. 9.8 (left) are replicas of the same objects across the layers.

Clearly, both classes of heterogeneity can be present at the same time. For instance, there may be different types of nodes (proteins, genes, metabolites, cells, . . .) and different types of relations between them (inhibition, activation, repression, expression, . . .). In most cases, a given type of relation can occur only between two specific types of nodes. But, given two types of nodes, there can be many (or no) types of relation connecting them. Again, some relations can be directed, others can be undirected. All these interactions can be of many-body types leading to hypergraphs.

As mentioned earlier, the fibration framework can easily and uniformly deal with this kind of scenario, that is, with graphs that have different types of nodes and/or edges. In the general fibration formalism, notions of node and arrow types are built into the definitions from the beginning.

More often than not, types are referred to as ‘colors’ in the graph-theoretic literature, and heterogeneous graphs are hence called (and represented as) node-colored and/or edge-colored graphs. We prefer to refrain from using the term ‘color’ here, because we are already using colors to refer to node equitable partitions, and having two types of colors would be confusing. Clearly non-heterogeneous graphs are a special case of heterogeneous graphs, where the set T contains only one element (all nodes and all edges have the same type).

What a type represents depends on the context, but nodes and edges with different types cannot be mixed by a symmetry: a gene cannot have the same role as a metabolite, an inhibition relation cannot have the same role as an activation. This ‘hard barrier’ imposed by typing in a heterogeneous world is formalized by requiring the relevant graph homomorphisms to preserve types.

Recall that, if G and H are heterogeneous, a graph homomorphism $f : G \rightarrow H$ is a graph homomorphism which further satisfies

$$\nu(x) = \nu(f(x)) \quad \eta(a) = \eta(f(a))$$

for all nodes $x \in V_G$ and edges $a \in E_G$. The idea behind this condition is quite obvious: typing provides an impassable wall between nodes and edges. Two nodes of different types cannot be mapped to each other, because they have a different nature. Similarly, two edges of different types will represent different types of interactions, so they cannot be related by a symmetry.

Once the notion of homomorphism has been generalized to the world of typed graphs, we can extend all the notions of fibrations, minimal fibration, etc. Everything holds *mutatis mutandis* in the heterogeneous world: we will have heterogeneous equitable partitions, the heterogeneous coarsest equitable partition, and algorithms to find such partitions just extend to the heterogeneous case by taking node and edges types into account at all stages. We discuss this further in Chapter 13 when talking about algorithms to find fibers.

9.5.1 Balanced coloring for a multiplex/multilayer weighted network

As an example, we generalize the balanced coloring to a brain multiplex. Neurons can communicate with each other via different signaling mechanisms and over a range of different

timescales. The result is a multiplex, multilayer weighted hypergraph of neuronal communication, which combines all the existing heterogeneities into one system. Understanding the symmetries of such a system poses significant challenges to standard graph theoretical approaches.

To account for the diversity of neuronal and synaptic properties in biological neuronal circuits, different types of links and nodes can be considered (Fig. 9.9). Typically, a neuron can simultaneously communicate via three routes: (1) chemical signaling via neurotransmitters, (2) electrical signaling via gap junctions, and (3) neuroendocrine wireless signaling via neuropeptides. Therefore, we consider three main classes of graphs c : $c = 1$ chemical synapses graph, $c = 2$ gap-junctions graph, and $c = 3$ neuromodulatory graph made of neuropeptide-mediated interactions. Each of these graphs will have different types of nodes and links, which reflect the diversity of the neuronal types, as well as of neuropeptides identities, gap junction properties, and diverse neuropeptide interactions.

For instance, the $c = 1$ synaptic class is subdivided into types t as excitatory, inhibitory, and neurotransmitter (NT) identity, i.e., glutamate, acetylcholine (ACh) and Gamma-Aminobutyric Acid (GABA). The wireless $c = 3$ graph is made of neuropeptide modulators. There is a large number of neuropeptide precursors encoded in the genome, potentially producing over 300 individual neuropeptides, and each neuron releases a unique cocktail of them.

Simply collapsing these interactions onto a single binary layer would result in an almost fully connected graph. At the other extreme, considering the combinatorial identity of each pairwise connection in this matrix would make nearly all of its elements unique. Both extremes limit any analysis of symmetrical structure or other graph theoretical measures. So, a symmetry analysis needs to be performed with care.

The model for balanced coloring which takes all this information into account is the following. The different multilayer graphs and strengths are taken into account by generalizing (9.4) to a multilayer/multiplex weighted matrix $w_{ki}^{c,t}$ which defines the strength of an edge ki of class $c = \{\text{chemical synapses, gap-junctions, neuropeptide connections}\}$ and type t as defined above. Each of these classes and types are balanced as:

$$i \sim j \iff \sum_{k \in D:ki \in E} w_{ki}^{c,t} = \sum_{r \in D:rj \in E} w_{rj}^{c,t} \quad \text{for each class } c \text{ and type } t. \quad (9.5)$$

9.6 Using typing to restrict the class of homomorphisms

Heterogeneity, expressed by typing, is often inherent in the nature of the system we are modeling. We can exploit heterogeneity to restrict the kind of symmetries we want to consider. Indeed, this is sometimes necessary to capture properties that otherwise would fail to be represented (Morone et al., 2020). As an example, we now consider a relevant problem related to nodes with no inputs.

Consider as an example the graph of Fig. 9.10. Here each node has either one or two inputs. Its coarsest equitable partition is the one shown in Fig. 9.11.

While this coloring can make sense in some contexts, in most cases we cannot simply postulate that all nodes with no inputs (1, 2 and 5 in this example) belong to the same class. Think, for instance, of a very large network with two such nodes very far apart, or even belonging to different components. In many situations it is much more natural to suppose that different nodes with no inputs all belong to different classes. One way to take care of this observation is to modify the color refinement algorithm in the following way (as suggested in (Morone et al., 2020)):

1. Identify all nodes with no inputs (nodes that receive signals only from themselves) from other nodes, and assign to each of them a different initial color.
2. Identify all nodes with no inputs from any node (including itself) and assign each of them its own separate color during every iteration of the algorithm.

While this solution certainly solves the problem, we can try to tackle the situation in a more principled and general way.

We can take the graph $G = (V, E)$ under consideration and suitably add types to its nodes as follows: nodes with no inputs are all assigned a different type, whereas nodes with one or more inputs are assigned the same type. This typing corresponds to the idea that nodes with no input will always be different from one another (and different from all the other nodes of the network, anyway), whereas we seek symmetries in the remaining nodes.

Formally, we use the following typing function for nodes:

$$\nu(x) = \begin{cases} x & \text{if } x \text{ has no inputs} \\ * & \text{otherwise.} \end{cases}$$

In this way, all nodes with one or more inputs have the same type, whereas nodes with no inputs all have different types. When executing any algorithm to compute balanced partitions on the heterogeneous graph obtained in this way, nodes with no inputs will be distinct.

Now, if we determine the coarsest equitable partition of this heterogeneous graph (see Section 13.6) we will obtain the coloring of Figure 9.12: now nodes 1, 2 and 5 have different colors (because we imposed them to have types). A consequence of this is that also 3 and 4 are no longer symmetrical to each other, because each of them receives an input from a different node (3 receives inputs from 1, and 4 receives inputs from 5). Nonetheless, some symmetry remains (6 and 7 have the same color, because they do receive inputs from symmetrical nodes).

We have discussed, for simplicity, only the case of a homogeneous graph on which we impose a typing that takes the no-input case into account. This idea can be easily extended to the typed case in a straightforward way. Nodes can be assigned the same color only if they have the same node type and the same number of input edges of each type. Not all such colorings are balanced—correspond to fibrations—but all balanced colorings must have this property since fibrations have the lifting property, and in this generalization, they must also preserve node and edge types.

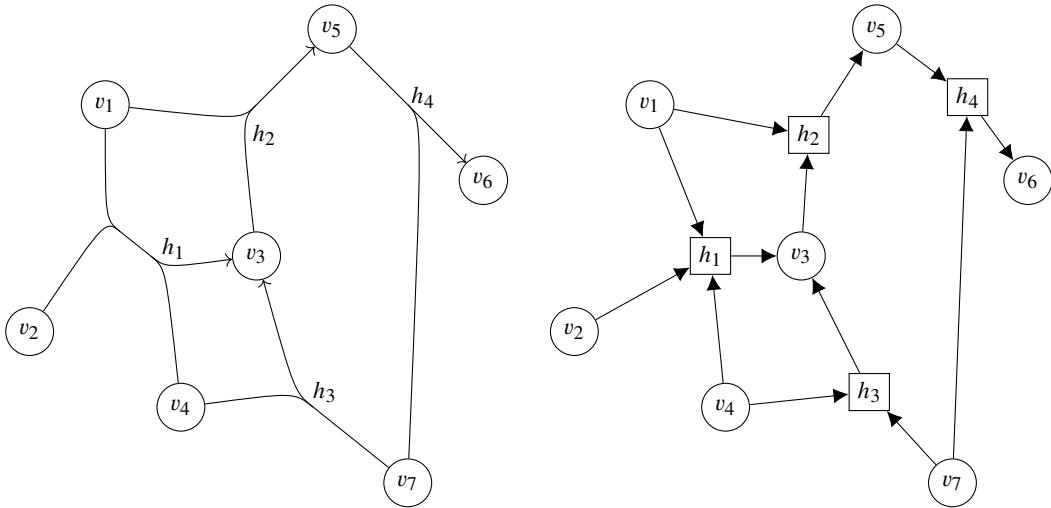


Fig. 9.3

Example of a directed hypergraph and a bipartite representation as a factor graph. *Left:* A directed hypergraph. For instance, nodes v_1, v_2, v_4 are inputs to the hyperedge h_1 that is the input function for node v_3 . *Right:* Its representation as a bipartite graph or factor graph. The square nodes are the hyperedges also called the factors. They represent the way to combine the inputs into a multi-body interaction in the input function of the output node. This represents the multi-body interaction in the ODE.

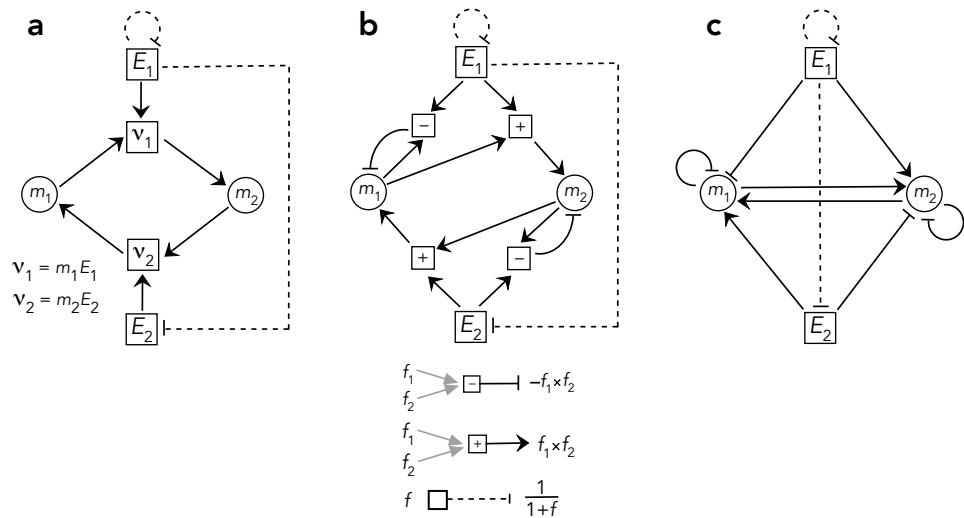
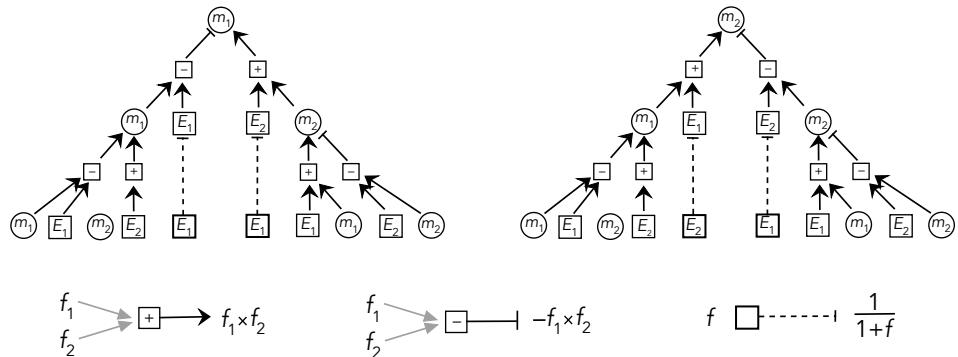
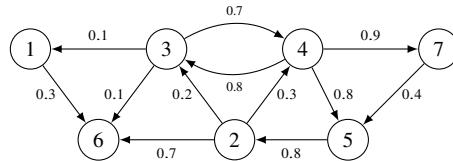


Fig. 9.4

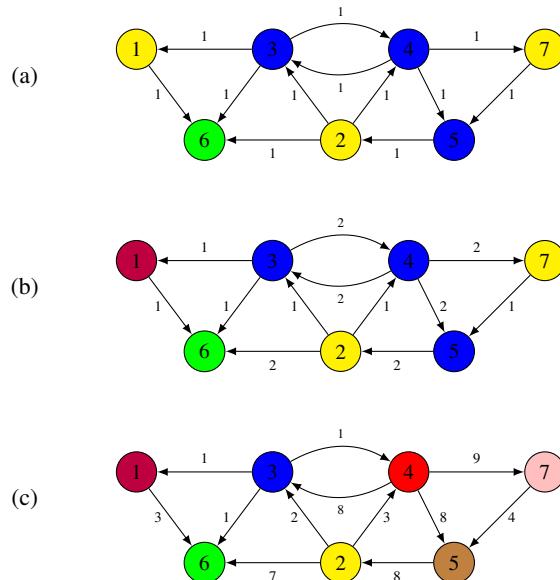
Comparison of different graphical representations of an ODE. (a) Biochemical representation for (9.1). (b) Hypergraph representation. (c) Admissible graph representation from Section 3.4 following the conventions of (Golubitsky and Stewart, 2023).

**Fig. 9.5**

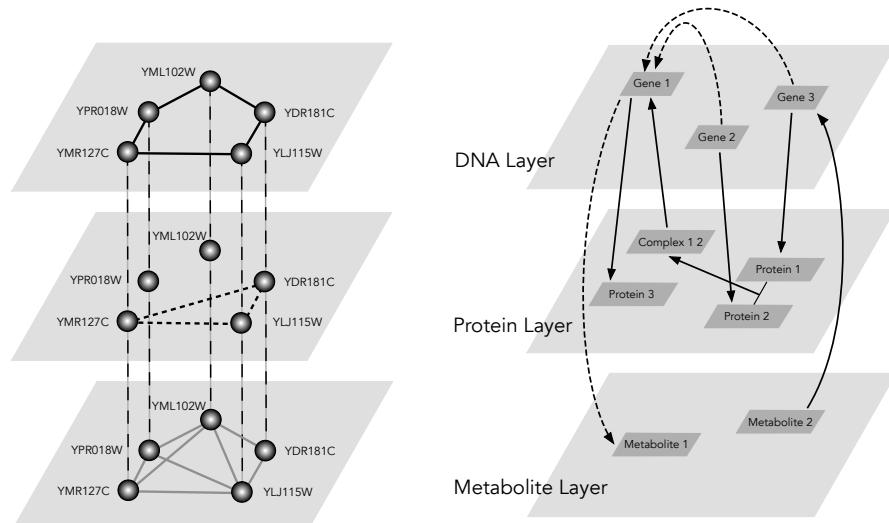
Input trees in the hypergraph. Input trees for nodes 1 and 2 in Fig. 9.4 are isomorphic, as would be expected from a fibration.

**Fig. 9.6**

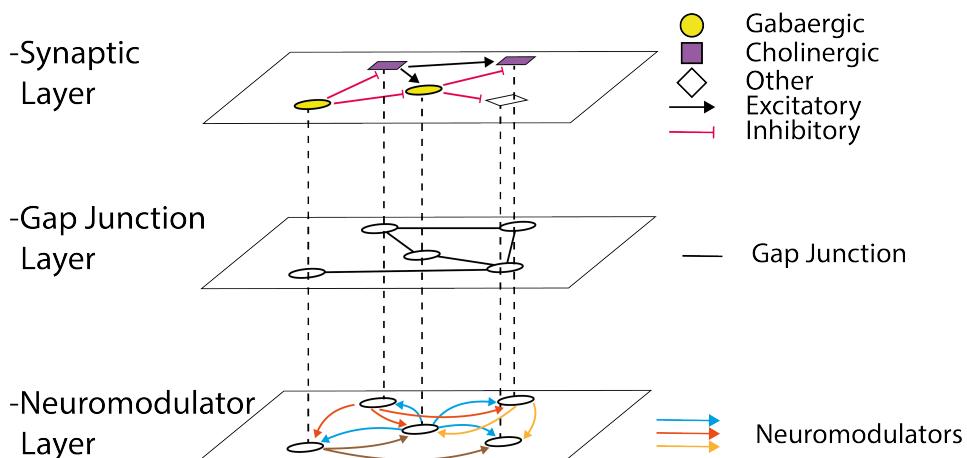
A weighted network. Each edge is labeled by its weight.

**Fig. 9.7**

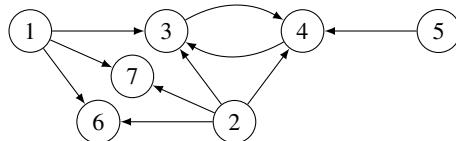
Possible discretizations of a weighted graph. Three typings obtained as different discretizations of the weighted network in Fig. 9.6, with the corresponding coarsest equitable partitions (balanced colorings).

**Fig. 9.8**

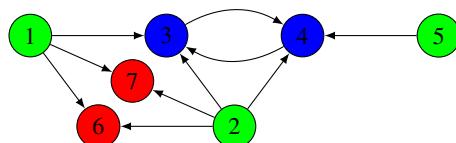
Heterogeneous graphs. *Left:* A multiplex, arc-heterogeneous (undirected) network of proteins: different layers (edge types) correspond to different types of interaction. *Right:* A metabolic network with multiple types of nodes forming a multilayer network. Figure reproduced from (Hammoud and Kramer, 2020).

**Fig. 9.9**

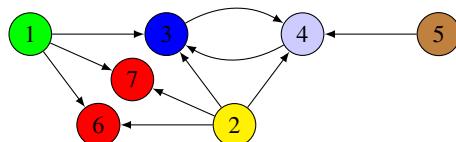
Multiplex/multilayer weighted brain hypergraph. The complexity of the brain network requires a heterogeneous model combining different types.

**Fig. 9.10**

Typing. A graph with many nodes having no inputs (1, 2 and 5).

**Fig. 9.11**

Minimal coloring. The coarsest equitable partition of the graph in Fig. 9.10.

**Fig. 9.12**

How to deal with nodes with no inputs. The coarsest equitable partition of the graph in Fig. 9.10 after preprocessing the nodes with no inputs. In many realistic cases, i.e., biological networks where nodes with no inputs corresponds to external sources, it makes more sense to consider them with different colors, rather than with the same colors as in the original definition of fibration.

The notions of fiber bundles, general fibrations, and graph fibrations are closely related, and further insight into fibrations is obtained when we compare them. In this chapter we elaborate on the links between these mathematical notions, seeking unification between the distinct areas in physics (where fiber bundles are abundant) and biology (where fibrations are abundant). Fiber bundles and fibrations are related but arise in different contexts: fibrations in homotopy theory and fiber bundles in topological geometry. The aim of the chapter is to sketch how these concepts are related, without providing many formal definitions. We give references for more formal treatments of these topics.

10.1 Fiber bundles

Fiber bundles go back to (Seifert, 1933), in a special case that contains the main ideas; it was later developed by numerous mathematicians, with the first general definition being given by Whitney (1935). ‘Fibration’ as a term in topology was introduced (with a slightly different name) by Serre (1951).

Fibrations became established as a general concept in the 1960s through the work of Serre and Grothendieck (1959) in algebraic geometry and category theory, inspired by the topological concept with the same name. Fibrations are maps similar to fiber bundles but more general (Steenrod, 2016; Husemoller, 1966). In this usage, the structure of the fibers in fiber bundles is constrained: often all fibers should be equivalent topologically with local Cartesian product structure. Moreover, each fiber can be equipped with a ‘structure group’ G that acts on it (in the same way for each fiber), in which case the fiber bundle is called a ‘ G -bundle’. Instead, in graph fibrations, the fibers can differ from one base point to another, which makes them more suitable for describing the great complexity of biology.

Fiber bundles and group theory provide a unified theoretical framework for modern physics. Fiber bundles describe the fundamental interactions in nature in a unified manner, forming the basis of the standard model of particle physics through gauge invariance of the principal bundle. Likewise, the requirement of invariance under local rotations in curved spacetime implies that general relativity is also a gauge invariant theory described by a fiber bundle, thus unifying all forces under the common notion of bundles.

To compare bundles and fibrations, it is convenient to first describe fiber bundles over manifolds and then generalize to the discrete case of fiber bundles and fibrations for graphs. Fiber bundles over manifolds appear when building a new manifold from the

topological product of two given manifolds. Products of topological spaces are very common in mathematics, since they allow the construction of larger spaces from smaller spaces.

Similarly to fibrations, fiber bundles consist of a base B and a fiber F which is copied to every point in the base to form the total space E . Both fiber bundles and fibrations capture the idea of one manifold (the fiber) being indexed or parametrized over another (the base) to form the total space of the bundle. Thus, the fiber of the bundle is a topological space which is parametrized by the base. From this point of view, fiber bundles appear to be similar to fibrations. Differences are discussed in the next subsection.

Fiber bundles can be seen as generalization of Cartesian products. We start by defining these product spaces, called trivial bundles; then we show how nontrivial bundles emerge by adding a ‘twist’ to the Cartesian product. We then repeat the same sequence of definitions for graphs, beginning with Cartesian graph products, generalizing to graph bundles, and ending with graph fibrations. With these definitions to hand, we then speculate on why fibrations, being a less restricted form of symmetry than groups, describe the natural symmetries of biological networks. We elaborate on why groups describe the inanimate world, while fibrations and groupoids describe the living world.

10.1.1 Cartesian products and trivial fiber bundles

The simplest (trivial) way to obtain a fiber bundle is by ‘multiplying’ two spaces using the Cartesian product.

Definition 10.1 **Cartesian product.** In set theory, the *Cartesian product* of two sets A and B is the set of all ordered pairs (a, b) : $A \times B = \{(a, b) \mid a \in A \text{ and } b \in B\}$.

If A and B are endowed with a topology, their product comes also equipped with a natural topology (Kelley, 1955).

A simple example of a Cartesian product is $\mathbb{R}^2 = \mathbb{R} \times \mathbb{R}$, which takes a line \mathbb{R} and places another line, also \mathbb{R} , at each point of the first line, to form a plane (in this case \mathbb{R} is endowed with a topology, which carries over to \mathbb{R}^2). Thus the two-dimensional space consists of two identical spaces \mathbb{R} that are combined to create the total space \mathbb{R}^2 , see Fig. 10.1 (top left). Likewise, we could consider the product of a line segment $(0, 1)$ and a circle (1-sphere \mathbb{S}^1) to form a cylinder.

These Cartesian products are special examples of fiber bundles: a fiber (line segment) is combined with the base (\mathbb{S}^1) to form the total space (essentially, a cylinder). The total space can be written as $E = B \times F$: here the total space is not just ‘locally a product’: it is *globally* a product. Cartesian products are *trivial* fiber bundles. In both cases \mathbb{R}^2 and the cylinder, a set of global coordinates is enough to specify any position in the total space. *Nontrivial* fiber bundles are generalizations of the global Cartesian product that look like a direct product locally, but not globally. Nontrivial fiber bundles are ‘twisted’ Cartesian products like the Möbius band, Fig. 10.1 (top right).

Before discussing nontrivial fiber bundles we make a detour to explain the importance of trivial fiber bundles in modern theoretical physics, by showing how trivial bundles describe

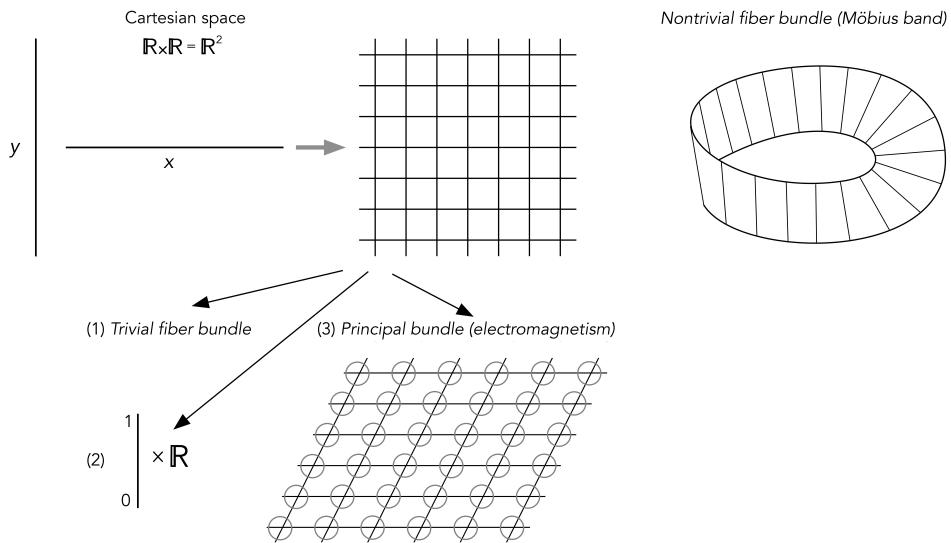


Fig. 10.1

Fiber bundles. Top left: the plane as an \mathbb{R} -bundle over \mathbb{R} . Top right: the Möbius band as an \mathbb{R} -bundle over the circle. Bottom: Electromagnetism as a circle bundle over \mathbb{R}^2 .

all fundamental interactions between elementary particles. Then we explain what we can learn from this detour to describe the biological world.

10.1.2 Principal bundles in physics

Fiber bundles can combine all kinds of topological spaces and structures. They are classified in terms of the kind of fiber space that is attached to the base manifold. For instance, when the fiber is a vector space, the resulting bundle is a vector bundle. A tangent bundle consists of the tangent spaces of a manifold. These fibers are important in the Lagrangian formulation of classical mechanics (Kibble and Berkshire, 2004). When the fiber is a Lie group we have a principal bundle (Steenrod, 2016).

Principal bundles play a fundamental role in physics by enabling the construction of a geometric theory of the fundamental forces. Interactions can be seen as fiber bundles over Minkowski spacetime, with a continuous Lie group as the fiber describing the particular gauge symmetry of the interaction. For instance, the quantum mechanical description of electromagnetism is understood as a fiber bundle over spacetime whose fibers have the gauge symmetry of rotations of a circle (Moriyasu, 1983). Figure 10.1 (bottom) pictures this fiber bundle by assigning a circle to every point of a two-dimensional spacetime base. The extra dimension given by the angle on the circle specifies the phase factor of the complex-valued wave function $\psi(x, t)$ describing electrons or charged particles moving in the manifold.

Gauge symmetry in quantum mechanics arises because observables are real numbers and therefore depend on the wave function according to the form $\psi^* \psi$. This creates a gauge

freedom in the definition of the wave function, since it can be rescaled by

$$\psi \rightarrow \tilde{\psi} = e^{i\alpha} \psi, \quad (10.1)$$

without any consequence for the observables. Indeed,

$$(e^{i\alpha}\psi)^*(e^{i\alpha}\psi) = e^{-i\alpha}e^{i\alpha}\psi^*\psi = \psi^*\psi$$

Mathematically, we can see these rotations as the action of the symmetry group $U(1)$ by rotations $e^{i\alpha}$ in the complex plane through an arbitrary angle α . The elements of this group are the fibers of the fiber bundle. The electromagnetic interaction between electrons arises when the phase factor is allowed to change along the base $\alpha(x)$ with $x \in B$. The gauge field that ‘glues’ the different circles together, while keeping the dynamics invariant under the local shifts of the phase factors, is the *connection* of the fiber bundle, which in this case is the electromagnetic potential A_μ . This gauge field determines how the position of an electron in the circle changes from a point in spacetime to its neighbor. The observed electromagnetic field or magnetic flux arises as the non-zero curvature of this trivial fiber bundle along closed trajectories (loops) in spacetime (Maldacena, 2015).

The symmetries of gauge theories are local in the sense that the parameter $\alpha(x)$ of the Lie group depends on the position x in the base. Global symmetry in this context is a gauge transformation with constant parameter α at every spacetime point, and it is analogous to a rigid rotation of the geometric coordinate system. A gauge transformation with a local parameter $\alpha(x)$ generates a local symmetry, and the gauge field of the physical interaction. However this ‘locality’ does not change the structure group defining the fiber bundle, which remains a trivial Cartesian product of the groups for the spacetime manifold and the manifold of the gauge group. Thus, even though the gauge symmetry is a local symmetry, the underlying fiber bundle remains of a trivial kind as the Cartesian product.

The mathematics of gauge theories was discovered by mathematicians as fiber bundles. Generalizing the gauge theory of electromagnetism, the theory of electroweak interactions was later discovered by considering the fibers as elements of the symmetry group $SU(2)$. Geometrically, this is described by a 3-sphere \mathbb{S}^3 whose elements are 2×2 matrices with unit determinant, and corresponds to the surface of a four-dimensional ball describing isospin space (Glashow, 1961). In similar fashion, the strong interaction of $SU(3)$ describes the color space of quantum chromodynamics (Greiner et al., 2007).

Going beyond the standard model, it is interesting that fibrations appeared in physics in the late 1980s through the quantum geometry of string theory, since these theories require the dimension of spacetime to be 10, while we observe only 4. A string theory vacuum decomposes the 10-dimensional space as $M_{10} = M_4 \times X$, where M_4 is Minkowski 4-dimensional space and X provides the extra 6 dimensions in a tightly curled-up manifold called a ‘Calabi–Yau fibration’ (Bao et al., 2020).

10.1.3 Nontrivial fiber bundles

Fiber bundles admit more interesting products than the trivial ones described above. Non-trivial fiber bundles generalize the Cartesian direct product of spaces to ‘twisted’ direct

products by describing structures that look like a trivial direct product locally, but not globally. This is analogous to a manifold, which generalizes Euclidean space by being locally, but not globally, Euclidean. In general, a nontrivial bundle is a local product of the base with the fiber.

The standard example of a nontrivial fiber bundle is the Möbius band, which is a ‘twisted’ Cartesian product of a line segment and a circle, Fig. 10.1 (top right). The Möbius band is created by attaching a copy of the line segment $[0, 1]$ to the circle \mathbb{S}^1 but in a nontrivial manner, by adding a 180° twist as we go around one revolution. The band can still be mapped onto the base \mathbb{S}^1 and, locally, still looks like the trivial cylinder. However, a point in the band cannot be described by global coordinates (the angle in \mathbb{S}^1 and the position in the line interval) as for the trivial global parametrization of the cylinder. Indeed, the Möbius band is not topologically equivalent to the cylinder; in particular it has a single boundary component and is non-orientable. The cyclic group $G = \mathbb{Z}_2$ is an example of the structure group of a fibration. It acts on each fiber by flipping it end to end, $x \mapsto 1 - x$. Traversing once round the base transforms the fiber in this manner.

Another example is a logarithmic spiral of Fig. 10.2, which can be projected radially onto the base \mathbb{S}^1 . For the full spiral, infinite in both directions, this is a fibration with discrete fibres homeomorphic to the integers \mathbb{Z} (the intersection of a radius with the spiral). Going once round the circle shifts the fiber by $n \mapsto n \pm 1$ depending on the direction (clockwise or counterclockwise), so the structure group is \mathbb{Z} under addition. Globally, the spiral and

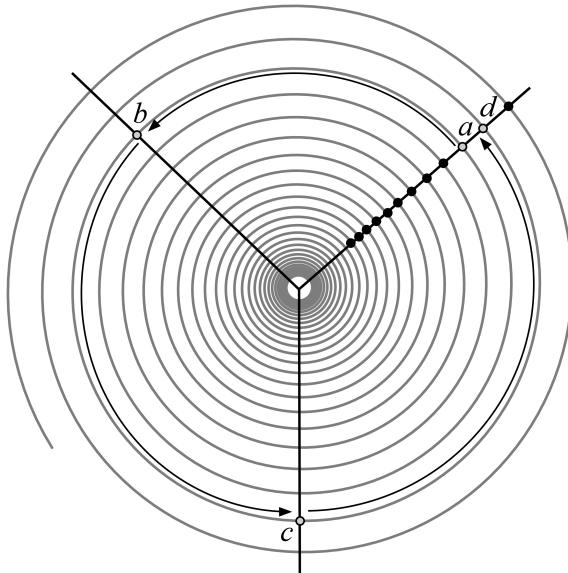


Fig. 10.2

Logarithmic spiral. A fiber bundle with base \mathbb{S}^1 and fiber \mathbb{Z} (black dots). One counterclockwise rotation around the base moves a typical point a on the fiber, via b and c , returning to the original fiber at $d = a + 1$. The entire fiber shifts one step outwards, from n to $n + 1$.

\mathbb{S}^1 are different. Without the shift up or down by 1, the figure would be an infinite set of concentric circles, a trivial \mathbb{Z} -bundle over \mathbb{S}^1 .

Connections

In terms of fiber bundles, a *connection* tells us how movement in the total space induces change along the fiber. A connection describes how to move from one fiber to another by describing how a quantity associated with a manifold changes as we move from one point to another by connecting neighboring fiber spaces. As discussed above, in the principal bundles of the standard model the connections are the gauge fields mediating the interactions, i.e., the fundamental forces. We come back to connections in Chapter 12 in the context of a financial model and its analogs for genetic and brain networks.

10.2 Fibrations in algebraic topology and category theory

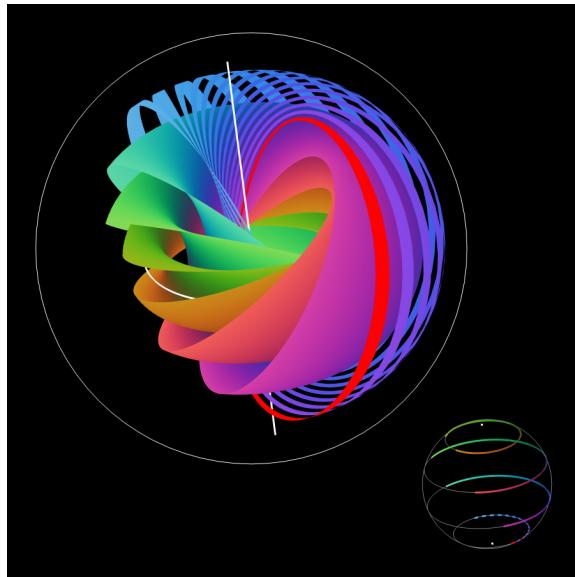
In algebraic geometry, the notion of fibration was introduced by Grothendieck (1959) in the context of category theory, inspired by the topological concept with the same name. But the topological idea, which developed into what arose in gauge theory, predates Grothendieck. In algebraic geometry, a fibration is a map from an algebraic variety to a lower-dimensional variety having some reasonable properties like lifting or surjectivity. In topology, a fibration is a further generalization of the concept of a fiber bundle. Fibrations in topology are denoted by $F \rightarrow E \rightarrow B$.

The common feature of bundles and fibrations is that both maps project a big space to a smaller one, and certain features transverse to the fiber are preserved. So each fiber is embedded in the big space in a nice way. These features are local neighborhoods for topology and input sets for graphs.

A distinctive feature of fiber bundles is that all fibers are homeomorphic to each other. In simpler terms, a fiber bundle is a map between fibers to the base, where all fibers are topologically equivalent to the same space. In a fibration, however, fibers need not necessarily be the same space. They still parametrize a topological space by a family of spaces, one for each point of the base, but there is a certain amount of freedom to choose different fibers along the base. In particular, in graph fibrations, the input tree needs not vary in any nice way from one fiber to its neighbors in the base.

Certainly, the condition that the fibers preserve local in-neighborhoods (that is, input sets) is the same for both fibrations and fibre bundles, although the technical implications are even more interesting since homeomorphisms do not have a direction, whereas combinatorial fibrations do not preserve out-neighborhoods, but only in-neighborhoods. This is the main difference that lets us apply fibrations to biological networks, which are usually directed networks.

One of the earliest examples of fibration in topology is the Hopf fibration, Fig. 10.3. This

**Fig. 10.3**

Hopf fibration. Ken Shoemake's picture of the Hopf fibration donated to the Hopf topology archive at purdue.pagaloo.com/d/edu/hopf.math (Johnson, 2012).

a locally trivial fibration since it is locally a product space. Thus, this fibration is technically a fiber bundle, referred to as a fibration in the literature for historical reasons. The Hopf fibration structure is denoted: $\mathbb{S}^1 \hookrightarrow \mathbb{S}^3 \xrightarrow{p} \mathbb{S}^2$. It has a local product structure since every point of \mathbb{S}^2 has some neighborhood U whose inverse image in \mathbb{S}^3 can be identified with the product of U and a circle: $p^{-1}(U) \cong U \times \mathbb{S}^1$. While this fibration is locally trivial, it is not a trivial fiber bundle, i.e., \mathbb{S}^3 is not a global product of \mathbb{S}^2 and \mathbb{S}^1 although locally it is indistinguishable from it. See Fig. 10.3.

Definitions

For the mathematically inclined reader, we end this section with topological definitions of fiber bundles and fibrations following (Cohen, 1998). This is one of the few expositions in the literature where both concepts are directly compared.

Definition 10.2 Fiber bundle in topology. A fiber bundle $F \rightarrow E : \pi \rightarrow B$ consists of a base space B , fiber space F , and total space E , together with a projection map $\pi : E \rightarrow B$. The total space E locally looks like the product of the base and fiber spaces. More precisely, every point $x \in B$ is contained in an open set $U \subset B$ such that there is a homeomorphism $\phi : \pi^{-1}(U) \rightarrow U \times F$. Further, $\pi \circ \phi^{-1}$ gives the projection onto the first factor of $U \times F$.

In classical homotopy theory, a fibration $p : E \rightarrow B$ is a continuous map between topological spaces such that the ‘homotopy lifting property’ holds with respect to all fiber spaces. The most basic property is that given a point $e \in E$ and a path $[0, 1] \rightarrow B$ in

Topology	→	Combinatorics
Cartesian product	→	Cartesian graph product
Fiber bundle	→	Cartesian graph bundle
Fibration	→	Graph fibration

Table 10.1: From topology to combinatorics.

B starting at $p(e)$, the path can be lifted to a path in E starting at e . A generalization of the notion of a fiber bundle due to Serre is simply a map that satisfies this type of lifting property. That is, a Serre fibration is a map between topological spaces $\varphi : E \rightarrow B$ that satisfies the ‘homotopy lifting property’. More generally, a Hurewicz fibration is a surjective, continuous map $p : E \rightarrow B$ that satisfies the homotopy lifting property for all spaces.

10.3 Graph products

We now generalize the topological concepts of Cartesian product, fiber bundle and fibrations to their combinatorial counterparts defined, respectively, as graph products, graph bundles, and graph fibrations. We explain the relations among these concepts, and what we can learn from them to describe biological networks. We use graph terminology, namely ‘vertex’ rather than ‘node’. Table 10.1 provides a short ‘dictionary’ for this analogy.

Generalizing the Cartesian product of sets, *graph products* are graphs whose vertex set is the Cartesian product of the vertex sets of its factors. In addition, they satisfy some condition on adjacency of vertices in the product, which depends only on adjacency in the factors. Graph products have been studied in discrete mathematics (Hammack et al., 2011), computer science (Jänicke et al., 2010) and even applied in biology (Fontana and Schuster, 1998).

Graphs can be ‘multiplied’ in a variety of ways. In a 1975 paper (Imrich and Izwicki, 1975), it was shown that there are exactly 256 possible products where adjacency in the product graph depends only on the adjacency properties of its factors. From these 256 products, only four standard products (Hammack et al., 2011) are typically used in the literature (Fig. 10.4). Below we follow the exposition in (Ostermeier, 2015; Hammack et al., 2011). Assuming two graphs $G_1 = (N_1, E_1)$ and $G_2 = (N_2, E_2)$ (in all cases $V(G_1 \times G_2) = V(G_1) \times V(G_2)$), two vertices (x_1, x_2) , (y_1, y_2) are adjacent (Fig. 10.4) in the following ways:

Definition 10.3 Four standard graph products:

- *Cartesian product*: $G_1 \square G_2$. Vertices are adjacent if (i) $(x_1, y_1) \in E(G_1)$ and $x_2 = y_2$, or (ii) $(x_2, y_2) \in E(G_2)$ and $x_1 = y_1$.
- *Direct product*: $G_1 \times G_2$. Vertices are adjacent if (iii) $(x_1, y_1) \in E(G_1)$ and $(x_2, y_2) \in E(G_2)$.

- *Strong product:* $G_1 \boxtimes G_2$ has edge set $E(G_1 \square G_2) \cup E(G_1 \times G_2)$, i.e., it is the union of the Cartesian product and the direct product.
- *Lexicographic product:* $G_1 \circ G_2$. Vertices are adjacent if they satisfy (ii) or if (iv) $(x_1, y_1) \in E(G_1)$.

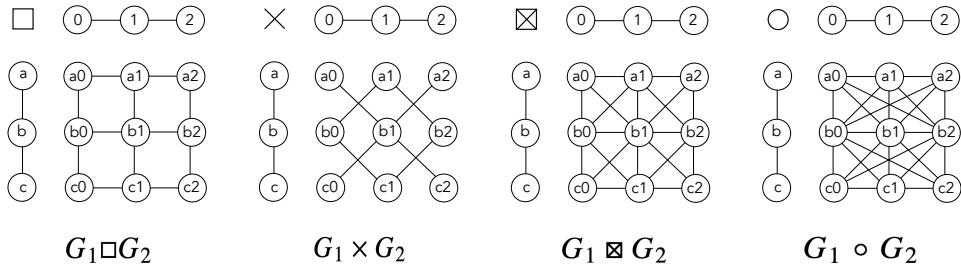


Fig. 10.4

Graph products. The four standard graph products.

10.4 Cartesian graph bundles

Graph bundles are a generalization of both Cartesian products and covering graphs. They can be seen as analogs for graphs of nontrivial fiber bundles like the Möbius band. While other graph products have been generalized to bundles, the most common one is the Cartesian product, so most graph bundles are Cartesian graph bundles.

Definition 10.4 **Cartesian graph bundle.** A *Cartesian graph bundle* is a graph G composed of a *fiber graph* F and a *base graph* B , together with a *graph map* $p : G \rightarrow B$ such that $p^{-1}(v) \subset F$ for each vertex $v \in V(B)$, and $p^{-1}(e) \subset K_2 \square F$ for each edge $e \in E(B)$. Here K_n is the complete graph with n vertices.

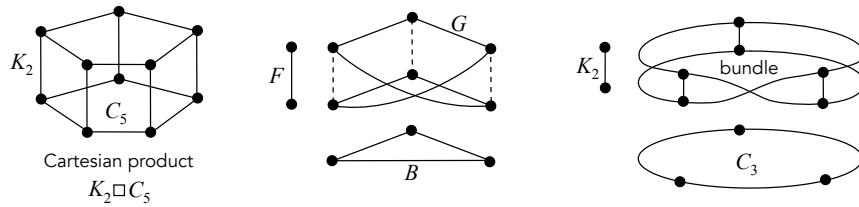


Fig. 10.5

Cartesian products. We show a graph bundle $K_2 \square C_5$ and a Cartesian graph bundle G over base graph $B = K_3$ with fiber $F = K_2$. The non-degenerate edges are highlighted with solid lines; the degenerate edges, which belong to the copies of the fiber, are depicted with dashed lines.

Figure 10.5 shows an example of the Cartesian graph bundle over the base graph K_3 with fiber K_2 , which is a graph representation of a Möbius band.

The concept of graph bundle is quite interesting since it has the properties of a fiber bundle and at the same time is discrete. Graph bundles were introduced in a seminal preprint by Pisanski and Vrabec (1983), which remains unpublished. Despite the importance of graph bundles as a bridge between topology and graphs, this preprint has enjoyed only mild attention from mathematicians, gathering only 36 citations in almost four decades.

10.5 Graph fibrations—history

The history of fibrations and their ramifications in various fields of algebraic geometry, topology, and other fields of mathematics, and beyond to computer science and systems theory is very rich and complex: more details can be found in Sebastiano Vigna's *Graph-Fibrations Home Page* (<http://vigna.di.unimi.it/fibrations/>). In the following, we just want to give an idea of the main lines of research involved.

The combinatorial definition of graph fibration used here was formalized in (Boldi and Vigna, 2002a) as a special case of Grothendieck's notion, obtained by looking at the free categories generated by the graphs (Mac Lane, 2013). More precisely, a graph morphism $\varphi : G \rightarrow H$ is a fibration if and only if the corresponding covariant functor $\varphi^* : G^* \rightarrow H^*$ is a Grothendieck fibration, where L^* is the free category generated by the graph L , also called a quiver (its objects and arrows are the nodes of and the paths of L , respectively).

Special instances of graph fibrations were rediscovered many times, under different names, in the literature before the notion was finally systematized in (Boldi and Vigna, 2002a):

- In the context of spectral graph theory, rear divisors (Cvetković et al., 1998) induce a graph fibration, thus providing a connection between fibrations, eigenvalues, eigenvectors and factorizations of the characteristic polynomial.
- A parallel line of research, also related to the computation of the characteristic polynomial, led to the introduction of equitable partitions (Schwenk, 1974), although apparently the connection with rear divisors went unnoticed.
- In the search for practical (albeit non-polynomial) techniques for the graph isomorphism problem, some efficient algorithms for the construction of the minimum base were discovered (Corneil and Gotlieb, 1970; Cardon and Crochemore, 1982), or even earlier (Unger, 1964). McKay (1981) made it explicit that the algorithm computes the coarsest (in our wording, minimal) equitable partition.
- In the computer science literature, the use of homomorphisms between graphs representing networks as a tool to prove impossibility results started with (Angluin, 1980); the necessity of extending those results to the directed case justified the shift from local isomorphisms (coverings) to fibrations.

Graph fibrations were introduced in computer science to prove (im)possibility results in distributed systems (Boldi and Vigna, 2001, 2002b). This study was inspired (see chapter

2.5 in (Boldi and Vigna, 2002a)) by an analysis of a network of processors (each node of this network executes the same algorithm). If all processors start in the same initial state, then existence of the fibration will cause all processors in the same fiber to stay in the same state, or in other words, to be synchronized. This is why fibrations have important consequences for information-processing networks and in computer science.

Fibrations have been also applied to study self-stabilization (Boldi and Vigna, 2002b) and to understand some topological properties of Google's ranking algorithm PageRank (Boldi et al., 2006). Finally, in the context of system dynamics, fibrations were proposed to study interconnections between subsystems and to generalize the group of symmetries (DeVille and Lerman, 2015b).

Armed with the analogies between fibrations, groupoids and fiber bundles in physics discussed above, in the next chapter we discuss the general applicability of fibrations to biology. Part II then provides empirical evidence of fibrations in biological networks.

In Section 6.7 we saw that fibration symmetries are more robust to changes in the network than automorphism symmetries, mainly because fibrations are local but automorphisms are global. We also saw that (in suitable circumstances) new outputs have less effect than new inputs. In this chapter we consider some biological examples where this principle applies: gene duplication and speciation. We argue that symmetry fibrations in biological networks may have been favored over symmetry groups by evolution due to their increased robustness (in an informal sense) and resilience under mutation events.

11.1 Homomorphisms for biological networks

We have seen that the concept of homomorphism (Definition 6.1) encompasses all the main maps that are important to characterize biological networks. We recall the definitions:

1. A (graph) homomorphism is a map that preserves the adjacency relation(s) for each node- and edge-type.
2. When the homomorphism is a bijection (injective and surjective) then it is an isomorphism. Isomorphisms induce an equivalence between input trees in fibers.
3. When the homomorphism is an isomorphism of the graph to itself, then it is an automorphism, i.e., a symmetry permutation of the graph.
4. When the homomorphism satisfies the lifting property, that is, when it collapses only nodes with isomorphic input trees, then it is a graph fibration.
5. Two nodes in the same fiber of a fibration always have isomorphic input trees.
6. A graph fibration that collapses all the available fibers into the minimal base of the graph is a minimal fibration or symmetry fibration. In a minimal fibration two nodes are in the same fiber *if and only if* they have isomorphic input trees.

A homomorphism can collapse any nodes in the network provided it preserves adjacency of nodes. A quotient map of a group of automorphisms collapses all nodes in the same orbit. A fibration collapses only nodes within each fiber (color classes for balanced colorings or nodes with isomorphic input trees). Every automorphism is a fibration, so all orbits are fibers, but not the converse.

To understand how these features of homomorphisms come together to explain the structure of biological networks, we use the example of graph G in Fig. 11.1a, which consists of the FFF to which we add a two-layer multilayer subgraph with a symmetry group. This graph contains four fibers (three are trivial and contain a single node, the other

contains two nodes) and two orbits, as indicated by the coloring. Orbit of the symmetry group are also fibers, but we want to make a distinction between the two types. Because of the automorphism $(4\ 5)(6\ 7)$, a quotient map (Definition 4.9) can be applied to this network to reduce it to $G/\text{Aut}(G)$ as in the figure.

From this quotient network, a fibration φ can be applied to reduce the fiber (in yellow) to its base as indicated. A direct fibration from G to B can be also applied since every automorphism is a fibration as shown (fibrations compose to give fibrations). We indicate this minimal fibration in the figure by $\varphi : G \rightarrow B$ collapsing the green, blue and yellow fibers.

Both orbits and fibers synchronize. However, we show in the next section that fibers may be favored over orbits under evolutionary pressure, because orbital synchrony is more vulnerable to duplication events and mutations.

Remark We can use $G/\text{Aut}(G)$ here because it gives the same quotient network as Definition 6.9. In general, however, $G/\text{Aut}(G)$ gives a different quotient, even for fibrations induced by automorphisms. See Section 4.2.3.

11.2 Duplication events break a group symmetry but not a fibration

Genetic evolution is in part driven by duplication of parts of the genome. Gene duplication events are shown schematically in the genetic network of Fig. 11.1b, where gene 9 is a duplicate of gene 3 and gene 8 is a duplication of gene 7. Both events duplicate the original gene, together with its expressed protein and promoter region. Therefore, the duplication event duplicates also the incoming link of the gene and the child gene inherits the full input tree of the parent gene. Thus, by definition, the duplication event preserves synchronization of the fiber. This demonstrates robustness of the fibration symmetry.

Remark *Here and in the rest of the book, the word ‘robust’ is used informally to indicate resilience in changing conditions. It is not used in the technical sense of Section 5.9.*

On the other hand, duplication of gene 7 into 8 breaks not only the orbit $\{6, 7\}$, but also the upstream orbit $\{4, 5\}$. Thus the entire symmetry group is removed by this single perturbation. This happens because of the global character of the symmetry group, which creates vulnerabilities that might propagate across the network structure.

In contrast, a fibration symmetry is more robust. Considering the fibration of the circuit, we observe synchronization of genes $\{4, 5\}$ and $\{6, 7, 8\}$. In this last case, the duplicated gene is added to the fiber, showing how the fibration not only preserves the synchronization state, but supports the addition of a new duplicated gene to a previous synchronized state without synchrony-breaking. This is shown in the base of Fig. 11.1b after the application of the fibration. On the other hand, the automorphism group predicts the appearance of a new partial synchrony in $\{7, 8\}$, but not $\{6, 7, 8\}$.

This vulnerability of automorphisms is also exemplified in the globally symmetric network example of Fig. 5.3a, which show how fragile group symmetries are: if the network evolves by adding just one extra node to the network, as in Fig. 5.3b, the global symmetry is completely broken. The automorphism group of the network in Fig. 5.3b now contains only the identity permutation. This occurs because automorphisms require strict arrangements of nodes and links to preserve the global structure of the network. Thus the addition of the outgoing link from node 3 breaks the synchronization, not only of the orbit $\{2, 3\}$, but also of $\{4, 5\}$, even though the added link does not connect to or emanate from this orbit.

These two cases exemplify the vulnerability of a synchronization state based on a global symmetry: a small perturbation in a part of the network can change the possible cluster synchrony patterns of the entire network, which often become very different. Potential synchronies can be destroyed—or even, sometimes, created. These changes can affect nodes located in another part of the network, not connected to the site of the added (or removed) edge or node.

A biological network with this global vulnerability cannot survive under evolutionary pressure. A robust evolving network requires a locally-symmetric fibration based on the input trees of symmetrically related nodes. Robustness induced by local symmetries guarantees stability of the phenotype, while at the same time providing flexibility to changes in network structure, suitable for evolvability of new phenotypes under natural selection.

11.2.1 Speciation events break a group symmetry but not a fibration

After a duplication event, speciation may occur by the addition of extra links, creating a new function for the duplicated gene, which no longer needs to be synchronized with the parent gene. This is exemplified in the network of Fig. 11.1c by the addition of node 10 to the duplicated gene 8 and the addition of gene 11 to the duplicated gene 9 in the fiber.

Again, the extra link to gene 10 breaks synchronization in the orbit $\{7, 8\}$, due to the new outgoing edge of node 8. Yet the synchronization predicted by the fiber $\{6, 7, 8\}$ remains intact after the speciation event, because the input trees of these nodes are not affected by the extra outgoing edge. Likewise, the addition of node 11 does not break the symmetry in the yellow fiber. The fibration ignores the break of synchronization predicted by the broken group orbits, and predicts the correct synchrony as observed in the base of the fibration, as in Fig. 11.1c. Because the sole condition for a network to be coherent and functional is symmetry in the inputs, but not in the outputs, there is complete freedom for the set of genes that connect the fiber to the external world to evolve, without altering the already functional fiber. These results show how a structure characterized by fibrations produces increased robustness of synchronization patterns compared to automorphisms, under duplication and divergence in the evolution of the genome.

11.2.2 Vulnerability of automorphisms under mutations

Figure 11.1d shows how a downstream mutation causing the loss of an edge breaks synchronization in both orbits. Again, in contrast, the fibration preserves synchronization in

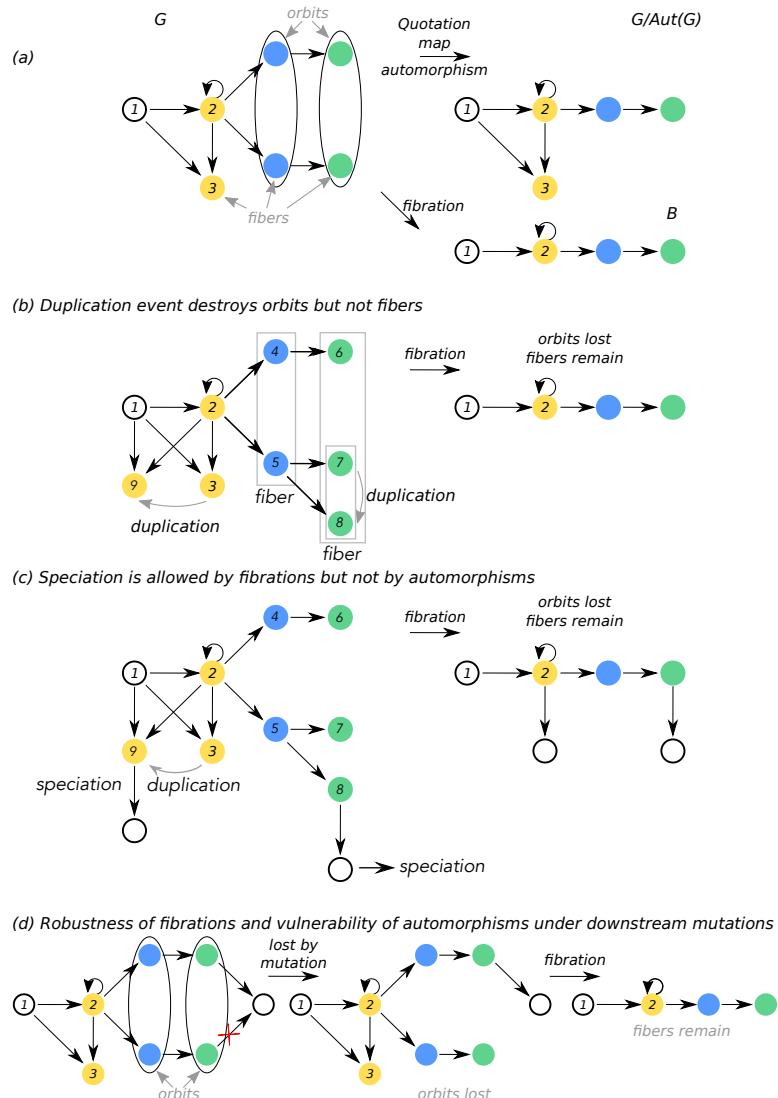


Fig. 11.1

Why fibrations are favored over automorphisms in biological networks. Fibration robustness versus automorphism vulnerability in biological networks. (a) A network and its fibration and quotient map under automorphism. (b) A generic gene duplication event destroys the automorphisms but fibration remains. (c) Fibrations permit speciation, automorphisms do not. (d) Fibrations are robust to downstream mutations; automorphisms are vulnerable.

both fibers, since in this case the fibration does not depend on the outgoing edges. The loss of synchrony in the group orbits is severe, since it happens upstream of the mutation, affecting genes that are not connected to the lost edge. This is another expression of the

vulnerability of a synchrony space based on a global symmetry. The fibration shown in the figure indicates that, instead, it is robust against this mutation.

11.3 Understanding ‘structure \rightsquigarrow phenotype’

In biological development, the ‘big data’ networks of molecular biology eventually manifest themselves in the phenotype of the organism. Early hopes that the DNA sequence of the genome could be ‘read’ to reveal the phenotype proved overoptimistic, and the way phenotype emerges from genotype remains enigmatic, despite numerous advances. It is also one of the central problems in biology.

In describing the arrow \rightsquigarrow :

$$\text{structure} \rightsquigarrow \text{phenotype} \quad (11.1)$$

we consider the structure to be given by the biological network, and the phenotype by its function and consider the simplest form of coherent functionality which is achieved by synchronization of the activity of the biological units: genes, metabolites, proteins, or neurons. We represent the space of biological functions as the quotient or base of the phenotype. In particular we evaluate the redundancy of the map \rightsquigarrow associated with the cardinalities of the fibers (Gromov, 2011).

Thus the existence of robust fibration symmetries has important consequences for the correlated dynamics of biological systems. The synchronization of neural and gene activity, as a consequence of network symmetry, is observed in patterns of correlated activity in biological networks, and this synchronization is largely independent of the dynamical model used to describe the activity of the nodes. Indeed, this is necessary, because precise models that match real biology with high accuracy are uncommon, unlike the situation in physics. Most biological models are phenomenological, and even those that include a lot of detail involve unknown parameters such as reaction rates and connection strengths.

11.3.1 Segregation versus integration of function

The collapse of the fibers by the symmetry fibration determines a partition of genes into segregated ‘sectors’ with a well defined functional interpretation. This suggests that the segregation of function in genetic networks could be a direct consequence of the symmetry fibration. This segregation of function occurs despite the integration of genes into a globally connected network. Therefore, the existence of fibers provides a possible mechanism to explain the conundrum of how segregation of functions can coexist with an integrated biological network. Thus, while a gene may be highly integrated by connections to other genes, the precise arrangement of links by fibrations still allows the genes to perform different functions according to the fiber controls, while remaining integrated into the whole network for coherent dynamics.

Thus, fibrations can explain not only the existence of modular structure in biological networks (Hartwell et al., 1999b), but also their robustness and evolvability. Loss of function

by breaking the symmetry of a fiber need not propagate to other fibers, because the fibration acts locally on the fibers. As a consequence, symmetry breaking in a fiber does not affect the symmetry of another fiber. If we associate each symmetry fiber with a function, then the failure of a given function does not affect the functions of other sectors. This theoretical prediction opens the possibility of experimental testing, by manipulating the activity of genes chosen to break the symmetry of a given fiber, and monitoring the resulting activity inside and outside the chosen fiber.

11.4 Axiomatization of biology

It is interesting to elaborate on how the biological symmetry of fibrations and groupoids emerges from a mathematical perspective in the hierarchy of fundamental algebraic structures. The axiomatic representation of the relevant abstract algebras involves four axioms: (i) Identity, (ii) Invertibility, (iii) Closure or composition law, and (iv) Associativity. A group is an algebraic structure whose elements satisfy the four axioms. In addition, we could add a fifth condition which is not considered to be a group axiom but is interesting in the context of earlier theories of quantum electrodynamics: (v) Commutativity, which leads to an Abelian group.

Elementary particles and their fundamental interactions are described by group symmetries following the four axioms in the non-Abelian gauge theory of the Standard Model. In addition the Abelian gauge theory of quantum electrodynamics also obeys commutativity (Weinberg, 1995).

Automorphisms represent global symmetries since they involve permutations of all nodes. In contrast, input isomorphisms and fibrations involve only local transformations that act on a subset of nodes and their input-sets. These isomorphisms are not necessarily closed under composition, and form a groupoid, which does not satisfy the axiom of closure. This suggests that a groupoid cannot be a symmetry of physics, since spacetime is composable (Weinberg, 1995). Removing one axiom from the algebraic structure of the universe brings huge freedom to the system—like looking at an entire new universe by dropping one fundamental axiom in the groups that are common in physics.

Groupoids are widely used in algebraic topology, because various topological constructions require specifying a base point in a topological space, and comparing what happens relative to that point with what happens relative to another point. The natural structure that results is a groupoid. Groupoids might be relevant in some places in physics, but probably not on the deep philosophical level of fundamental laws.

Outside their uses in certain branches of mathematics, groupoids are seldom encountered. They are not widely used in the way that groups are used in physics, with powerful tools such as group representations, irreducible representations, and so on. The groupoid formalism does not provide a similarly powerful mathematical technique that can be widely applied. Instead, the most important information that can be extracted from groupoids is the concepts they lead to. In this case, these are things like balanced colorings and fibrations.

With Einstein's hat on: the big group symmetries of physics arise because the laws of

physics are the same at every point in spacetime. If the laws of physics changed from one region to another, there would not be a symmetry group for those laws... but there could be a symmetry groupoid. This symmetry groupoid would encode the answer to the following question: if two physicists were at different places, would they observe the same laws of physics? If the laws of physics are consistent with a symmetry group, the answer would be ‘yes’. But if the laws of physics are consistent with a symmetry groupoid, then physicists would see the same physics only when they are in the same connected component of the groupoid (in the sense of composition), with the same local vertex symmetries.

We argue that the axiom of closure may be an ‘accidental’ constraint rather than a fundamental one, and thus unnecessary for a living system. We call it accidental because this axiom holds only for special symmetries of the system, i.e. automorphisms, whereas in general it is not valid for the majority of symmetries, which constitute the symmetry groupoid.

Symmetry groupoid transformations identify clusters of synchronized balanced-colored nodes. Symmetry groupoid transformations do not impose any restriction on where the outputs are sent. That is, nodes in a synchronized groupoid cluster are free to send their outputs to any node in the network that does not lie in that cluster. (This can break up other clusters, however.) The outputs of nodes in a group orbit—a synchronized group cluster—are restricted by the global structure of the whole network.

In Section 6.7 we compared the rigidity of automorphisms with the flexibility of fibration symmetries. We concluded that although the situation is subtle, the focus on input sets rather than outputs as well, combined with the local nature of fibration symmetries, groupoid allows biological networks to undergo changes such as gene duplication and speciation without much change in synchrony patterns. The distinction between inputs and outputs requires a directed graph, such as a genetic or neuronal network, where the distinction between the input and output sets of a node is meaningful. This might be why, in many biological systems, directionality of an effect has been favored. This may have given a groupoid circuit a subtle evolutionary advantage, allowing it to emerge more commonly than group circuits. In an undirected network, like the gap junction connections in the neural network of *C. elegans* studied in Section 23.4, the evolutionary advantage of fibration over automorphisms is diminished. Indeed, these gap junctions form a group like the physical bonds in molecules.

The ability of different nodes in the circuit to perform the same function—may give a groupoid structure an evolutionary advantage over group structures with a concomitant increase of robustness. If matter is a group and life is a groupoid, so to speak, then a key feature of genetic networks might be that their biological information is encoded in the symmetries of the directed genetic signals that genes receive from each other, rather than in the identity of the gene itself.

11.4.1 Speculations on axioms for biology and the Erlangen Program

Nearly all physical systems have emerged from higher to lower symmetries following a series of spontaneous ‘symmetry breaking’ events (Weinberg, 1995; Georgi, 2018; Landau

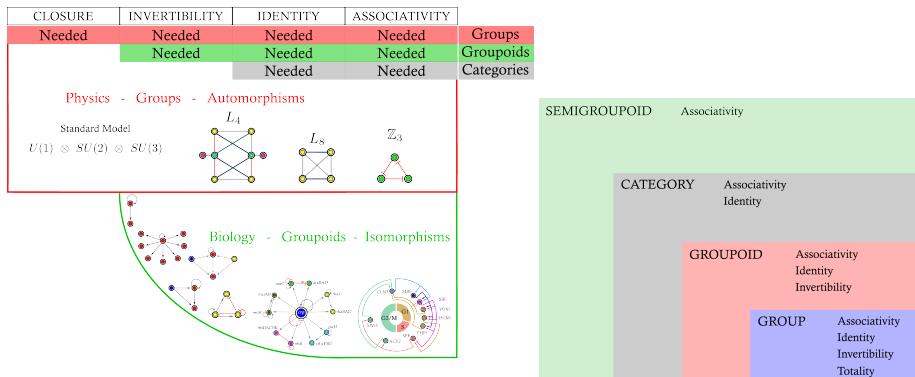


Fig. 11.2

Evolution of groupoids in the hierarchy of algebraic structures. *Left:* Biological networks may have emerged by an 'algebra breaking' event by relaxing the axiom of closure (law of composition) from the more stringent structures characterized by physical symmetry groups. Symmetry groups shown are $U(1) \otimes SU(2) \otimes SU(3)$ of the Standard Model (Weinberg, 1995), the locomotors L_4 and L_8 (Section 23.4), and repressor \mathbb{Z}_3 (Elowitz and Leibler, 2000). The symmetry groupoid branch represents the building blocks and composite groupoids found in biological networks. *Right:* Nested hierarchy of algebraic structures considered in this book. Starting from the most restrictive algebra, Abelian Groups, we relax one axiom at a time to obtain: Abelian groups → Non-Abelian groups → Groupoids → Categories → Semigroupoids.

and Lifshitz, 1977), without abandoning the four algebraic axioms of non-abelian groups which were hard-wired in the fabric of spacetime from its origin. From a very speculative viewpoint: the emergence of biological groupoids and life may have involved a more drastic change in the appropriate algebraic structure itself, through a process that we might metaphorically describe as 'algebra breaking' from physical groups by abandoning the axiom of composition to obtain the less restrictive algebraic structure of groupoids (Fig. 11.2). If evolution from matter to life can be seen as an algebra-breaking event, from non-abelian groups to groupoids by relaxing one axiom, it is natural to consider what other algebraic structures could emerge from groupoids. Relaxing the axiom of invertibility leads to a category of irreversible processes (Eilenberg and MacLane, 1945) which satisfies only associativity and the existence of the identity and further completes a series of transitions from Groups → Groupoids → Categories.

Of course, there are more ways to generalize an algebraic structure than by dropping individual axioms, and alternative axiom systems can define the same structure. But the hierarchy of structures here is especially significant. Biology, due to its complexity, necessitates less constrained laws than physics does. Physics is static, so groups can describe all molecules, but biology is evolving, so it needs less rigid algebras. There is an analogy with the physics of phase transitions: a system can have full symmetry, like a liquid, but after a phase transition, a lower symmetry state is found in the ground state. Thus a solid breaks the

symmetry of the liquid and brings up a new state of matter. All symmetries in physics are eventually broken. To go from biology to physics, a more fundamental property is broken: the fundamental algebraic structure that describes biology as opposed to physics.

Such an algebraic view of natural processes may lead to a ‘geometrization’ or ‘axiomatization’ of biology, analogous to Felix Klein’s 1928 ‘Erlangen Program’ (Klein, 1872), that could successfully organize scattered biological knowledge in a systematic way.

A geometrization program of biology (more details in Chapter 12) would follow the systematic application of symmetry principles to all biological networks. This program could provide a classification of the elementary building blocks of living networks from first principles to open the way to predict function and dynamics from the symmetries of transcriptomes, proteomes, metabolomes, connectomes and beyond. A geometrization of biology would imply a systematic classification of all building blocks of biological networks from the bottom up using first principles based on symmetries (in analogy to geometry). We will attempt the first steps of such a program in Part II of this book.

It is as though this axiomatization of symmetry in biology might be as fundamental in biology as in physics, but with a twist. As noted by Bourbaki (1994) p.47, the law of composition is the most primitive notion in the axiomatization of algebra, to the extent that it has been inseparable from the evolution of algebra since the first calculations with whole numbers. By liberating natural systems from the strong requirements of the composition law imposed by physics and its symmetry groups, a gentler symmetry has emerged, and with it, the circuits that sustain life.

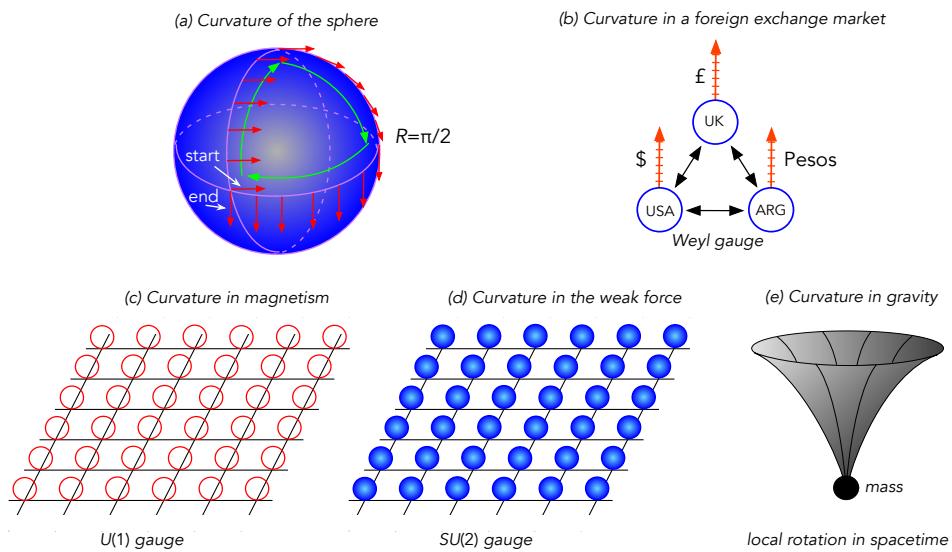
In this chapter, we sketch a geometrization of biological networks (fibrations) by analogy with the geometrization of physics (fiber bundles). There, geometrization is performed by introducing the curvature of a fiber bundle, using the concept of parallel transport along closed loops. In physics, this procedure leads to the association of the local curvature (e.g., of spacetime) with the fundamental forces of nature (e.g., gravity) via the connection of the fiber bundle. We first introduce the concept of curvature from a classical geometric point of view and generalize the concept to the modern view of gauge theory in theoretical physics. We then describe genetic networks as a fibration in a closed loop with nonzero curvature. As it turns out, an analogous gauge theory for biological networks is the dilation symmetry group used by Weyl (1919) in his early (but failed) attempt to unify electromagnetism and general relativity. Weyl gauge theory has been applied to a toy model of financial markets by Ilinski (2001) and Young (1999), and reviewed by Maldacena (2015). Borrowing heavily from Maldacena's essay, we follow this simple analogy between gauge theories in physics and financial markets to introduce geometry into biology through the more general framework of fibrations.

12.1 Curvature and connection in geometry

Section 11.4 was speculative and informal. In this chapter, we attempt to tighten the reasoning and clarify some of the ideas it leads to, though this discussion remains speculative. One such idea is curvature.

The curvature of a surface is measured by transporting a stick (a vector in a vector field) along a closed loop on the surface while attempting to keep the stick straight along the trajectory. When the orientation of the stick at the end of the transport across the loop is different from the initial orientation, the surface has non-zero curvature. This construction is realized by the rules of parallel transport of the stick, and can be easily visualized on the 2-sphere \mathbb{S}^2 (Fig. 12.1a). Parallel transport of a vector tangent to the sphere occurs when we transport the vector along a great circle of the sphere (i.e., the intersection of the sphere and a plane that passes through the center of the sphere), in such a way that the vector tangent to the great circle moves always tangent towards another point in the great circle. Intuitively, we transport a stick while keeping it always parallel to the local tangent plane of the surface. The definition of parallel transport also applies to any path, irrespective of whether it follows a great circle or not, by decomposing the path into a series of small steps along great circles.

In this setting, curvature arises when the vector moving under parallel transport along a

**Fig. 12.1**

Curvature and connections. (a) Curvature of the sphere is measured by the parallel transport of a vector in a closed circuit (green arrows from ‘start’ to ‘end’). (b) Curvature in a toy model of an exchange market. (c) Curvature in QED. (d) Curvature in electroweak fiber bundle. (e) The curvature of spacetime in general relativity.

closed loop returns to the initial location pointing in a different direction. This difference in direction is proportional to the curvature of the surface.

For instance, the parallel transport of a vector along a loop composed of the spherical triangle defined by three great circles of the 2-sphere depicted in Fig. 12.1a results in a rotation by $\pi/2$ compared with the initial vector. This is a consequence of the curvature of the sphere, and the difference in orientation along a closed loop encodes this curvature. In contrast, when we parallel transport a vector on a flat surface (which can be viewed, heuristically, as a sphere with infinite radius, hence zero curvature) the initial and final orientations of the vectors are the same.

This idea gave rise to the concept of a connection, which arose historically as a way to define differentiation on a curved spherical surface. Here the comparison of two arbitrary tangent vectors is not well-defined, because the coordinate system changes from point to point along the surface. To capture this change of coordinate system along a curve in the surface, a ‘connection’ field is defined, which produces parallel transport along the curve. The notion of a connection then captures this change of coordinate systems, which lets us compare two vectors on the sphere, and it makes precise the idea of transporting a vector in a parallel and consistent manner along the curved surface. When two vectors at two points along the curved surface are compared by the connection field, then, provided there is no difference between the vectors, those vectors are the result of parallel transport along the surface, and the covariant difference along the curve is zero. On the contrary, if the vectors

are not connected by parallel transport, there is a non-zero covariant difference along the curve.

The area enclosed by the curve of the closed loop, S , times the curvature associated with the connection, \mathcal{R} , is the differential between the vectors as they are parallel-transported along the loop. In symbols,

$$\Delta = \mathcal{R}S. \quad (12.1)$$

For the closed loop on the sphere in Fig. 12.1a, the curvature of the sphere of radius R is $\mathcal{R} = R^{-2}$, and the area enclosed by the loop is $S = \pi R^2/2$. Therefore, the difference is $\Delta = \pi/2$. This agrees with the previous discussion.

12.2 Curvature and connection in physics

Curvature can be defined locally (near a given point), and there are several distinct notions of curvature. Gaussian curvature is local: at a given point, it is the product of the two principal curvatures. Local curvature is a metric property, preserved by isometries (transformations that preserve distances *within the surface*). Thus a cylinder has zero local curvature at every point, just like a plane, and indeed the cylinder can be unrolled to form a plane without distorting distances.)

In contrast, total curvature is global: it is the integral of the Gaussian curvature over the entire surface. It is more general than the actual physical geometry of the object, characterized by such quantities as angle, metric, or length. There is no need for a metric to define the geometry because curvature is a property of the topological class of the shape. Thus if two surfaces are homeomorphic to one another (same topology), they will have the same total curvature, even if they have different shapes. This is the Gauss-Bonnet Theorem, and it is valid for compact surfaces without boundary (do Carmo, 1992).

Likewise, curvature can be defined for a fiber bundle, characterizing its local geometry through a connection. Indeed, the 2-sphere forms the base of a fiber bundle with two-dimensional tangent planes (fibers) attached to every point of the base to form the total space. The curvature of parallel transport across a closed loop in a fiber bundle characterizes its geometry in the same way as the curvature of the 2-sphere.

The geometrization of physics is achieved through the curvature of the fiber bundle, and fiber bundle geometry can be extended to biological fibrations. To set up a geometrization of biology, we begin with an intriguing analog of the geometrization of physics in the financial sector. Then we develop this analogy more specifically for the salient features of the geometrization of modern physics in general relativity and the standard model. Finally, we extend the analogy to biology.

12.3 Curvature and connection in a financial fiber bundle

Physical fiber bundles consist of Minkowski spacetime as the base together with different fibers associated with gauge invariant groups that represent the fundamental forces of nature: gravity, electroweak and strong forces, as given by general relativity and the standard model of physics, respectively. These forces correspond to local rotational invariances of general relativity and the quantum gauge theories with fiber symmetry groups $U(1) \times SU(2) \times SU(3)$. However, trying to understand biology through an analogy with the gauge invariances of gravity and quantum mechanics of the standard model could be quite traumatic and threatening.

Fortunately, this is not necessary. The main ingredients for understanding curvature are already present in the simplest gauge-invariant theory, represented by the multiplicative group of dilation introduced by Weyl (1919) to describe electromagnetism. Gauge invariance is the term used by physicists to talk about what mathematicians call the structure group of the fiber bundle (Moriyasu, 1983). Thus, gauge theory and fiber bundles are two ways to talk about the same theory, the first for physicists, the second for mathematicians. Modern quantum gauge theories are Yang-Mills theories from the 1950s, but the term ‘gauge invariance’ originated in a seminal (but failed) attempt of Weyl (1919, 1929b,a); Moriyasu (1983) to unify gravity and electromagnetism, using the dilation group as the gauge invariance.

It turns out that the dilation group describes also the gauge symmetry of a toy model of a foreign exchange market which maps to classical electromagnetism. This happens because the value of one currency in terms of another depends on the exchange rate, which is the ratio of the numerical amounts concerned. If a currency is revalued by government intervention (in the example below, by defining a new currency with three zeros removed from the denomination) and all exchange rates are adjusted to reflect this change (multiply or divide the relevant exchange rate by 1000) then the underlying financial transaction are effectively unchanged.

This analogy between foreign exchange and gauge theory was developed by Ilinski (2001) and Young (1999), and popularized in (Maldacena, 2015; Schwichtenberg, 2019). We follow the exposition of Maldacena, where it was shown that Weyl gauge invariance describes an economic toy model of speculators exchanging currencies between different countries. This is a very simple example of a gauge theory, and it is a genuine analog of classical electromagnetism in the absence of charges. It turns out that curvature is the key concept that unifies all these disparate systems.

Gauge invariance is present in the economic model of foreign exchange of Ilinski (2001). Imagine a toy model of foreign exchange among three countries with their own currency, USA (USD), UK (£), and Argentina (Pesos). Speculators can freely exchange currency among these three countries according to the exchange rates defined by the respective central banks. The base of the fiber bundle is the network of three countries shown in Fig. 12.1b. The network has undirected links since speculators can exchange money either



Fig. 12.2

Argentinian monetary gauge symmetry in action (Maldacena, 2015). In the 1980s the Argentina Central Bank removed three zeros from the currency denomination and called the new currency ‘Australes’. This change had no noticeable impact on economics, since the prices of all goods adjusted to the new denominations instantly, revealing a gauge symmetry in the market. In fact, the Central Bank did not even find it necessary to print new notes, and a simple stamp sufficed to apply the gauge transformation. However, this change in currency led to an adjustment of the exchange rates with other countries, leading to a ‘force’ (connection) with those countries. When these connections are non-zero, a curvature emerges in the market, which is a manifestation of arbitrage opportunities round a closed circuit, analogous to magnetic flux in electromagnetism.

way. The system is a fiber bundle with gauge symmetry, and we will show that it displays curvature analogous to magnetic flux.

We attach a fiber representing the money space $F = \mathbb{R}_+ = [0, +\infty)$ to each country in the base. The fiber measures the amount of money in that particular country, or the price of any good in the currency of that country. This means that an element of $f \in F$ can be any real positive number; it expresses the value of an asset or good in the currency of the country. This fiber has gauge symmetry: the units of the currency can be changed arbitrarily without any consequence for the economy. For example, an apple in Argentina can cost 1 Peso or 1 trillion Pesos according to the units of the currency set by the central bank, and, in principle, nobody would notice the difference. Using either currency would just imply changing the numbers of zeros on the banknotes, with no real-life consequences.

In fact, because of recurrent systemic inflation over the last 60 years, Argentina has changed the unit of currency over time by removing zeros from its bills (not to be confused with devaluing the currency). Since the 1960s this Argentinian gauge symmetry has been applied many times, removing a total of 12 zeros. One Peso of today is equivalent to 1 trillion Pesos of the 1960s. One of these gauge transformations is shown in Fig. 12.2. It occurred in 1985 when the Central Bank removed three zeros from the Pesos bill to transform it into a new currency called ‘Australes’ at a rate of 1 Austral = 1,000 Pesos.

Only a toy model of the economy that neglects psychological effects or transaction costs can claim that removing zeros from the currency has *no* consequences for the markets, so this gauge symmetry is never perfect in reality. However, it is a useful didactic example of a gauge transformation that leaves the system (approximately) invariant.

Such a transformation corresponds to the action of the dilation group, $GL^+(1, \mathbb{R}_+)$: this is the one-dimensional general linear group over the real numbers (with positive determinant), and it consists of all multiplications by any positive real number. This group is the structure group G of the fiber bundle. In this case, G is the group of functions $g : \mathbb{R}_+ \rightarrow \mathbb{R}_+$ that acts by multiplying $x \in \mathbb{R}_+$ by a constant $\lambda(g) \in \mathbb{R}_+$: $g(x) = \lambda(g)x$.

While the change of currency unit has no consequence for the internal economy, there is a catch: the Central Banks and foreign exchanges need to update their exchange rates with Argentina to keep the comparison of prices of goods intact, i.e., the dynamics of speculators exchanging money among the countries should remain invariant under the gauge transformation.

For instance, imagine that before the adjustment to Australes, £1 was equivalent to 1,000 pesos, i.e., exchange rate $R_{\text{£} \rightarrow \text{pesos}} = 1,000 \text{ pesos}/\text{£}1$, and the exchange rate to dollars was $R_{\text{pesos} \rightarrow \text{USD}} = 500 \text{ pesos}/1 \text{ USD}$. After the transformation of the Argentinian currency 1,000 Pesos \rightarrow 1 Australes, the exchange rates with neighboring countries need to be adjusted by the central banks to $1\text{£} = 1 \text{ Australes}$ and $1 \text{ Australes} = 2 \text{ USD}$, so that the dynamics of money exchange across the countries remains unaffected by the change, and we can safely (covariantly) compare the value of goods in all currencies (Fig. 12.3).

These exchange rates are exactly the connections of the financial fiber bundle, and they are the glue that lets us compare how much a currency is worth in terms of another (e.g., how much gold is worth in one country with respect to another one) in the same way that the covariant derivative in the sphere lets us compare vectors moving along the curved surface of the sphere. Thus the connection provides the rule for adjusting the coordinate systems in fibers associated with different neighboring points of the base.

To define the curvature of the fiber bundle we perform a parallel transport of the currencies crossing from country to country along a closed loop of the base, e.g., from USA \rightarrow UK \rightarrow ARG \rightarrow USA, Fig. 12.3. This is the financial analogous of the parallel transport in the sphere of Fig. 12.1a.

For this, we define the third connection between USA and UK with a rate $R_{\text{USD} \rightarrow \text{£}} = 1\text{£}/2\text{USD}$. A speculator going around the loop could take one dollar in USA and parallel transport that dollar across the loop and obtain around the circuit: $1 \text{ USD} \rightarrow 0.5\text{£} \rightarrow 500 \text{ pesos} \rightarrow 1 \text{ USD}$. In detail: $1 \text{ USD} \times R_{\text{USD} \rightarrow \text{£}} \times R_{\text{£} \rightarrow \text{pesos}} \times R_{\text{pesos} \rightarrow \text{USD}} = 1 \text{ USD} \times \text{£}1/2\text{USD} \times 1,000 \text{ pesos}/\text{£} \times 1\text{pesos}/500 \text{ USD} = 1 \text{ USD}$. This is exactly the same dollar at the end of the loop when the speculator is back in the USA, Fig. 12.3a. That is, the gain or *arbitrage* around the loop is zero. This means that the curvature of the fiber bundle defined by these particular connection exchange rates is zero: it is a ‘flat space’, as flat as a plane, and the speculators would gain no money by traversing this financial circuit.

In practice, speculators go around financial circuits to find opportunities for arbitrage: an opportunity to make money out of thin air (risk-free profit) from fluctuations in the market. In the case of currency arbitrage, it is the opportunity to make money from a misadjustment in exchange rates. For instance, imagine that the Argentinian Central Bank is not quick enough to adjust their exchange rate after the adjustment of currency and deviate (even slightly) from the zero curvature connection discussed above. In doing so, instead they set the exchange rate with USA as $R_{\text{pesos} \rightarrow \text{USD}} = 1\text{USD}/100\text{USD}$. Then, going around parallel transporting money in the same circuit starting with 1 USD in USA, the speculator

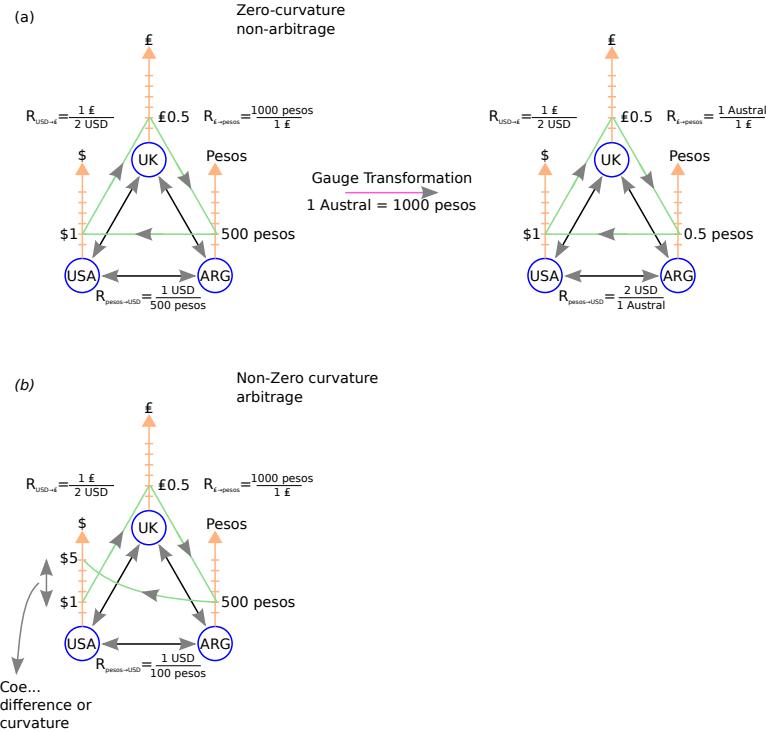


Fig. 12.3

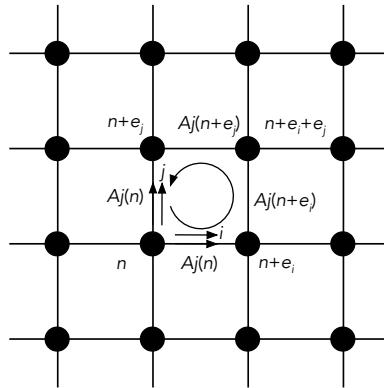
Financial curvature. (a) Zero curvature in an equilibrium market with no arbitrage. (b) Non-zero curvature in a financial market with arbitrage opportunities to make money out of thin air by just going around the loop.

will take advantage end up with a $5\times$ gain, Fig. 12.3b. This gain corresponds, geometrically, to the curvature of the ‘surface’ enclosed by the circuit, by analogy with the sphere. Thus, whenever we move in a loop and end up in a different state from the initial one, we say that the space of the base is curved, and this curvature defines the geometry of the fiber bundle.

12.3.1 From finance to physics

To calculate the curvature, we identify countries as points in a lattice, which for convenience we consider to be a two-dimensional lattice as in Fig. 12.4. Lattice gauge theories in physics (Creutz, 1983) are generalizations of continuum four-dimensional spacetime in physics to a lattice base. Lattice gauge theories were introduced by Wilson (1974) to describe non-Abelian gauge theories of the strong interaction on a space-time lattice (Creutz, 1983). A lattice formulation of the fiber bundle is shown in the discrete spaces of the financial market (Fig. 12.3) and electromagnetism (Fig. 12.4).

We now consider the lattice model of Fig. 12.4, following (Maldacena, 2015). In physics, the logarithm is introduced to work with the Lie algebra rather than the Lie group, which is a more complicated object. We therefore work with the logarithm of the exchange rates,

**Fig. 12.4**

Lattice gauge theory. Example for a financial market and electromagnetism.

which plays the role of the magnetic potential $A_i(\vec{n})$, at position \vec{n} in a two dimensional lattice $d = 2$, and $i = 1, \dots, d$, according to the formula

$$R_i(\vec{n}) = e^{A_i(\vec{n})}. \quad (12.2)$$

The gauge transformation at position \vec{n} multiplies the currency of country at \vec{n} and can be written as $f(\vec{n}) = e^{\varepsilon(\vec{n})}$. Under the action of this gauge, the connections with neighboring countries at direction \vec{e}_i (see Fig. 12.4) are changed as:

$$R_i(\vec{n}) \longrightarrow \frac{f(\vec{n} + \vec{e}_i)}{f(\vec{n})} R_i(\vec{n}), \quad (12.3)$$

or

$$A_i(\vec{n}) \longrightarrow A_i(\vec{n}) + \varepsilon_i(\vec{n} + \vec{e}_i) - \varepsilon_i(\vec{n}). \quad (12.4)$$

The curvature F_{ij} is then obtained as the traders go around the elementary circuit shown in Fig. 12.4 that starts and finishes at (i, j) . The gain (or loss) of money in such an elementary financial circuit is:

$$\text{gain} = R_i(\vec{n}) R_j(\vec{n} + \vec{e}_i) \frac{1}{R_i(\vec{n} + \vec{e}_j)} \frac{1}{R_j(\vec{n})} = e^{F_{ij}(\vec{n})}, \quad (12.5)$$

or

$$F_{ij}(\vec{n}) = A_j(\vec{n} + \vec{e}_i) - A_j(\vec{n}) - [A_i(\vec{n} + \vec{e}_j) - A_i(\vec{n})]. \quad (12.6)$$

Extending the base space to four-dimensional spacetime with elements $x = x^\mu$, with $\mu = 0, 1, 2, 3$, and taking the continuum limit (lattice spacing in Fig. 12.4 to zero, see (Maldacena, 2015)), we obtain the curvature tensor characterizing the financial market geometry:

$$\mathcal{F}_{\mu\nu}(x) = \frac{\partial A_\nu}{\partial x^\mu} - \frac{\partial A_\mu}{\partial x^\nu}, \quad (12.7)$$

with the transformation of the connection (12.4) given by

$$A_\mu(x) \longrightarrow A_\mu(x) + \frac{\partial}{\partial x^\mu} \varepsilon(x). \quad (12.8)$$

This notion of curvature is analogous to magnetic flux in electromagnetism. It is invariant under the change of currency, and provides the gain or loss of money around the elementary speculative circuit.

12.4 Curvature and connection in the electromagnetic fiber bundle

We have seen that speculators can make (or lose) money by traversing a closed financial circuit if there is positive (negative) curvature; moreover, this curvature is analogous to the manifestation of magnetic flux in a gauge theory of electromagnetism. This analogy can be further extended by incorporating the transport of goods, like silver or gold, analogous to moving electrons. This now introduces a particle field, in addition to the gauge field of the interaction, but we stop the analogy here since we already have enough elements to move to the physical fiber bundle of electromagnetism, and then to jump to describe curvature in a genetic fibration.

In the electrodynamic analogy, the exchange rates are the magnetic potentials defined along the edges in the base space. The speculators are the electrons, and in the presence of non-zero magnetic fields they move in circles, like speculators in search of arbitrage opportunities. The total gain obtained from the curvature of the financial circuit is now the total magnetic flux enclosed by the closed loop. Thus an electron moving in a loop because of the magnetic field is analogous to a speculator moving in a financial circuit, getting richer because of the non-zero curvature.

This gauge invariance in classical electromagnetism is a manifestation of the non-observability of A_μ ; that is, only the electric and magnetic fields can produce observable effects (modulo the Aharonov–Bohm effect, see below). Briefly, the electromagnetic four-potential is $A^\mu = (\phi/c, \mathbf{A})$, where ϕ is the scalar potential and \mathbf{A} is the vector potential. Maxwell's equations are invariant under the transformation:

$$\mathbf{A} \longrightarrow \mathbf{A} + \nabla \Lambda, \quad \phi \longrightarrow \phi + \frac{1}{c} \frac{\partial \Lambda}{\partial t}, \quad (12.9)$$

where Λ is an arbitrary function (Moriyasu, 1983). In terms of the four-potential:

$$A_\mu(\vec{x}) \longrightarrow A_\mu(\vec{x}) + \frac{\partial}{\partial x^\mu} \Lambda(x), \quad (12.10)$$

which is the same transformation as in the financial model (12.8). In other words, the connection field in Maxwell's equations A_μ transforms under a gauge transformation in a fiber bundle with a multiplicative group, offering the same geometric analogy as in the financial model. Furthermore, in electromagnetism the curvature tensor $\mathcal{F}_{\mu\nu}$ is the electromagnetic field tensor, whose components are expressed in terms of the electric and magnetic fields, meaning that the observable electric and magnetic fields define the curvature of the bundle. We see then that a complete geometric theory is possible for both the financial model and classical electromagnetism. The physical electric and magnetic fields are the elements of the curvature tensor, the four-potential is the connection and

'spiritus' of the interaction, and the dynamical equations (Maxwell's) are invariant under the symmetry transformation.

The dilation gauge invariance employed in (12.10) is the invariance employed by Weyl in his failed attempt to unify electromagnetism and gravity by enlarging the gauge group of general relativity of local rotational symmetries in spacetime (Lorentz transformations) to include also dilation symmetry of vectors. Indeed, since the invariance in electromagnetism (12.10) and the dilation group (12.8) are the same, it was natural to interpret the four-potential as the connection of the dilation gauge symmetry of electromagnetism.

However, as was pointed out by Einstein and others, this symmetry is incompatible with special relativity. Scale invariance of the dilation group implies that there is no well defined scale of physical vectors. However, the length of the four-momentum is $\|p\| = -E^2/c^2 + \mathbf{p} \cdot \mathbf{p} = -m^2c^2$, so the mass of a particle breaks dilation symmetry. Local gauge invariance survived, and 18 years after Weyl, a quantum framework showed that the correct gauge invariance of electromagnetism is not the dilation group, but the one-parameter $U(1)$ symmetry group of rotations in the complex plane. This symmetry group is also a multiplicative group. Indeed, there are only two options to choose from if we are interested in a one-parameter multiplication group: $U(1)$ and multiplication by positive real numbers (dilation group) (Ilinski, 2001).

Electromagnetism is based on the gauge invariance specified by the symmetry group $U(1)$, which corresponds to the symmetry of rotations of a circle. This group is associated with redundancy of the phase factor in the wave function of the electron, Eq. (10.1). This is captured by a fiber bundle of spacetime where at every spacetime point a circle is attached that defines the phase factor (Fig. 12.1c). That is, the complex plane with coordinates $z = x + iy$ is the fiber, and the structure group is the group of multiplications by a complex number $e^{i\alpha}$ of unit modulus: $U(1)$. Thus, the phase α is the variable to be gauged. As the electron moves along the base of the fiber bundle, the magnetic field, defined by the connection or magnetic potential A_μ determines how the phase factor changes from circle to circle for two neighboring points in the base, Fig. 12.1c.

Gauge theories are relevant to all the forces in theoretical physics. The weak force arises from the symmetry of a sphere at each point of spacetime, Fig. 12.1d. The dynamical field or wave function Ψ is an isospin doublet where the two components refer to the quantum amplitudes measuring the state of a particle that is either a proton or a neutron, rather than a single complex number. This is approximate since it assumes that we turn off electromagnetism, so we cannot distinguish a proton from a neutron by the electric charge; it also ignores small mass differences. These isospin states are indistinguishable: the doublet can be transformed as $\Psi(x) \rightarrow \Psi'(x) = T\Psi(x)$ where T is a 2×2 matrix for isospin and it is an element of a Lie group. In general, the groups are non-Abelian, as opposed to the Abelian group $U(1)$. When the gauge is $U(1) \times SU(2)$ we obtain the QED theory of the electroweak interaction (Weinberg, 1995), while adding the gauge group $SU(3)$ completes the standard model with the strong interaction (Glashow, 1961; Greiner et al., 2007).

12.5 Curvature and connection in gravity

In the *Scholium Generale* of the *Principia Mathematica* (Newton, 1833), Isaac Newton laments his own failure: '*Hactenus phænomena cælorum & maris nostri per vim gravitatis exposui, sed causam gravitatis nondum assignavi.*' ('Thus far I have explained the phenomena of the heavens and our sea by the force of gravity, but I have not yet assigned a cause to gravity.') In the '*Hypotheses non fingo*' paragraph that has attracted much scholarly attention, Newton concludes, regarding the origin of gravity: '*Sed haec paucis exponi non possunt; neque adest sufficiens copia experimentorum, quibus leges actionum hujus spiritus accurate determinari & monstrari debent.*' ('But these things cannot be explained in a few words; furthermore, there is not a sufficient number of experiments to determine and demonstrate accurately the laws governing the actions of this spirit.')

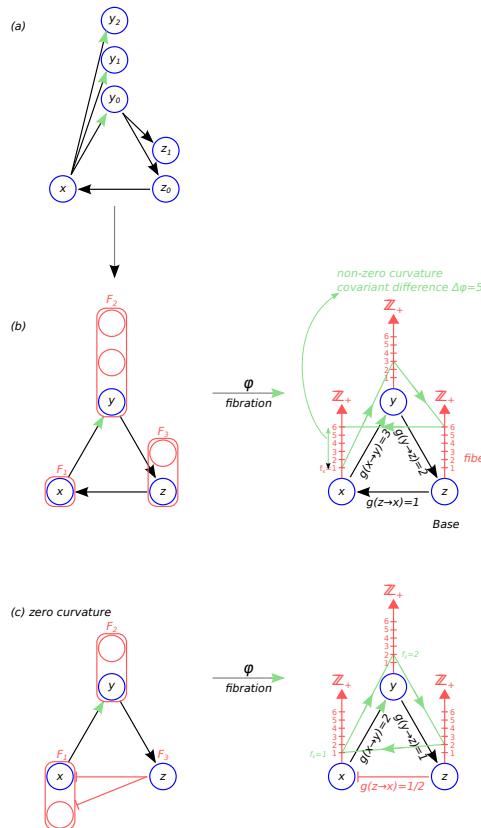
Humanity had to wait 249 years until Einstein read the last chapter of Newton's *Principia* and disentangled the meaning of this '*spiritus*' through his geometric theory of gravitation, showing that massive objects influence each other through the curvature of empty spacetime, which is itself caused by those objects. This concept of curvature was then applied to all fundamental forces and particles in the standard model through the curvature of the fiber bundle, providing the intellectual framework to think about them all with the same kind of concept underlying all interactions. In the next section, we generalize curvature and connection in terms of biological fibrations, thus extending Einstein's geometric view to genetic (and other biological) interactions.

We have seen that the core concept of force arises from seemingly superfluous redundancies in gauging the fibers of the system, like changing the currency units in economics or the phase factor of the wave function in quantum mechanics. Yet these redundancies are highly nontrivial, because they trigger changes in the interactions between neighboring elements in the base through the connection. The interaction is then the adjustment of the connection that is needed to compare elements along a path in the base.

The parallel transport of elements in the fiber along two different paths with the same initial and end points can be different. The curvature of the fiber bundle captures this difference, and non-zero curvature indicates the existence of a physical force in nature, or the possibility of risk-free financial gain or arbitrage in exchange markets. This non-zero curvature then triggers the motion around the loop of fundamental particles (with charge) and speculators, respectively. The non-zero curvature is the '*spiritus*' of the force; thus answering Newton's final scholium question. We now show that this '*spiritus*' is also present in the transfer of genetic information in the fibration formalism.

12.6 Curvature and connection in biology

After this lengthy but informative motivation, we reach the main point of this chapter: to generalize the curvature concept to genetic fibrations.

**Fig. 12.5**

Curvature in a genetic fibration. (a) Genetic network with a closed loop to define the genetic curvature. (b) The fibers represent redundancies in the network. These redundancies can be eliminated by collapsing all genes in a fiber to its base gene of the fibration. We show the connections along the edges (gauge fields) and the elements in the fibers. The cross-sections going around the loop give rise to a positive curvature. The TFs going around the circuit get ‘richer’, as in the financial model with arbitrage. (c) Zero curvature in a cycle with repressor links. The curvature of this graph is zero since the cross-sections going around the loop end up with the same value as the original one. The curvature is zero, and the TFs do not get ‘richer’ or ‘poorer’, by analogy with an equilibrium economic model with no arbitrage, or the absence of gravity in a flat spacetime.

12.6.1 Flow of information in a genetic network: the genetic ‘spiritus’

We consider a transcriptional regulatory network (genetic network) with a closed loop as shown in Fig. 12.5a. We explain the concepts in this simple fibration structure with three fibers, F_1, F_2 and F_3 (Fig. 12.5b), forming a \mathbb{Z}_3 -symmetric ring at the base. The ring

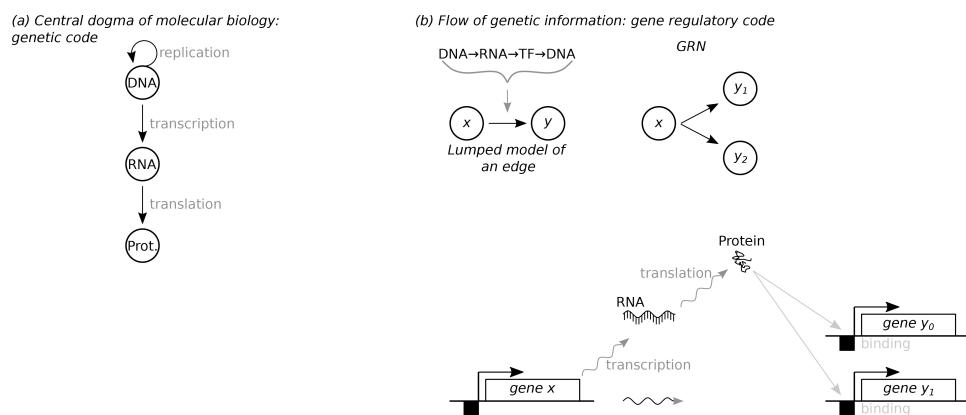


Fig. 12.6

Flow of information in a genetic network: the genetic ‘spiritus’. (a) Central dogma of molecular biology (Crick, 1970). Rare processes are excluded. (b) Interpretation of the information flow in the genetic network. A gene is expressed (first transcribed into a long mRNA molecule, which is then translated into a protein (transcription factor, TF), which, upon folding, acquires a structure that allows the TF to bind to the promoter site, controlling the expression of another gene. Thus, the source gene sends a ‘TF messenger’ to activate or repress the target gene, which represents the transmission of genetic information across the network.

is the closed loop or cycle necessary to develop curvature. We will develop the concept of genetic curvature using this simple example, which can then be generalized to more complex structures cycles.

To facilitate the analogy with financial and electromagnetic fiber bundles, we interpret the genetic network in terms of message-passing along its edges. A link in a transcriptional network represents regulatory messages that are dynamically sent from a source gene to a target gene via a transcription factor, which acts as the messenger, Fig. 12.6. Each arrow in the network represents a message sent from a source gene to a target gene, by analogy with links between countries when speculators exchange currencies, or spacetime links in a lattice model of electromagnetism or gravity. Here we simplify the biological regulation process to the case of bacteria. Eukaryotes have more complex forms of regulation, e.g., splicing and TF binding to distance enhancers that regulate genes via DNA looping, which also need to be taken into consideration; however, they do not alter the main picture.

This flow of information in biology is understood in terms of Francis Crick’s ‘central dogma’ of molecular biology: ‘DNA makes RNA (transcription), and RNA makes protein (translation)’; Fig. 12.6a (Watson and Crick, 1958; Crick, 1970, 1988). Information about the function of the protein is encoded in the sequence of amino acid residues in the protein, which in turn is encoded in the sequence of nucleotides in DNA. The dogma is the framework for understanding the transfer of this information from DNA, encoded in the A, G, T, C sequence, to the structure and function of the protein.

The DNA sequence of the source gene is first read by an RNA polymerase to transcribe

the gene's A, G, T, C code into RNA codes A, G, U, C, where the RNA polymerase binds the target DNA at the promoter region at the start of the gene. In doing so, the segment of DNA representing the source gene is copied to RNA. The RNA is then read by the ribosome and translated into an amino acid chain (or polypeptide), which then turns spontaneously into a folded protein whose structure determines its function in the cell. The encoding of each amino acid of the protein is done through the genetic code in groups of three nucleotides, called codons. The combined transcription → translation → folding process is called gene expression. To this pipeline, a loop DNA → DNA is added to represent the self-replication process necessary for life (Fig. 12.6a).

However, the transfer of information does not stop with the folded protein. To realize its function, the protein binds with other biopolymers and molecules, thus extending the information transmission through the network of binding among proteins, DNA, metabolites and other small molecules. Recall that a transcription factor (TF) is a protein that binds to another DNA segment. It thus acts as a messenger that controls the transcription rate of the target gene by binding to its promoter region (Fig. 12.6b).

This information flow is not restricted to two interacting genes, but it is transferred to different regions within the network that are accessible through the connected pathways. Likewise, a link can represent any kind of regulation, either repression or activation of any strength, directed or bidirectional, and can describe any dynamical law of genetic regulation, such as a Hill input function. This genetic information travels through the network and contains the whole history of information transmitted through all possible pathways that reach a given gene. This process of communication of information between different genes is formalized by the input tree of the gene.

Thus, a link in a transcriptional network does not represent a static physical link, like a molecular bond. Links in a genetic network represent regulatory messages that are sent dynamically from a source gene to a target gene via the transcription factor which acts as a messenger of genetic information by binding to the promoter region of the target gene to turn its activity on and off. The causal interaction from source → target extends the flow of genetic information of the central dogma from $\text{DNA}_{\text{source}} \rightarrow \text{mRNA} \rightarrow \text{protein (TF)} \rightarrow \text{DNA}_{\text{target}}$, by adding the last step where the protein is a transcription factor that regulates the transcription rate of the target gene (Fig. 12.6b). This flow of genetic information is the biological analogous of Newton's 'spiritus'.

12.6.2 Curvature and connection in a genetic network

We have extended the causal flow of genetic information from source gene → target gene using the transcription factor as the messenger. In this framework, DNA makes RNA, RNA makes proteins (TFs), TF regulates DNA to make fibrations, and fibrations make the phenotype.

We now define this genetic interaction in terms of a connection between the fibers attached to the base of the fibration. We follow the definition of a connection in the fiber bundle, defined in Sections 12.3 and 12.4 for the financial and electromagnetic model, using a discrete base. These lattices can be generalized to a graph to describe genetic interactions.

We follow the generalization of the continuum space to a discrete base from (Ilinski, 2001), which is particularly suited to describe curvature and connection in graph fibrations.

The discrete base of pointlike countries in Fig. 12.3 now becomes a graph of genes interacting according to Fig. 12.5a. In this case we have three genes in the base arranged in a \mathbb{Z}_3 ring: $B = \{x, y, z\}$. To each gene in the base we assign a fiber F_x, F_y, F_z with a given number of genes $|F_x| = 1, |F_y| = 3, |F_z| = 2$. The genes inside each fiber are all equivalent, since they have the same input tree, and are therefore redundant from a dynamical point of view. The analogous fiber bundle is now represented as in Fig. 12.5b, where we exemplify the base graph and the fibers attached to each node of the base. The fibration collapses this total space to give the base.

By repeated duplication, we can add any number of genes to the fibers without changing the dynamics, and this represents the gauge invariance of the system analogous to changing the units of the currency in the financial model, or the phase factor in quantum mechanics. From this perspective, the fiber is defined as the number of genes associated to each gene at the base. An element f of the fiber F_i indicates that there are f redundant genes at the base i . Thus the fiber is the space \mathbb{Z}_+ , the positive integers:

$$F_i = \mathbb{Z}_+ \equiv [1, +\infty) \quad (i = x, y, z). \quad (12.11)$$

The number $|F_i|$ is fixed for a given genetic realization of the cell, but it can be changed arbitrarily with no changes to the dynamics of the genes (aside from changes to some connection strengths, see below). An element f of the fiber can increase by one by an evolutionary duplication event (this will be treated in more detail in Section 18.3, see Fig. 18.2 and Section 11.2) by creating one gene in the fiber F_i . Likewise, a mutation or speciation can remove a gene from a fiber. These changes are performed by a transformation of the structure group G associated to the fiber, in similar way as the $GL(1, \mathbb{R}_+)$ group generates the changes of arbitrary units of currency in the financial fiber bundle, or the transformations of $U(1)$ generate gauge rotations in the phase factor of the wave function in the electromagnetic fiber bundle.

The structure group acting on the fibers is the dilation (or contraction) group over \mathbb{Z}_+ corresponding to maps g that multiply the number of genes in the fiber by an arbitrary number. For a fixed $f \in F_i$ this is a ‘one-parameter semigroup’ (with identity, that is, a ‘monoid’) $\mathbf{G} = \mathbb{Q}_+ = \{q \in \mathbb{Q} : q > 0\}$. This semigroup acts as transformations of the fiber, with an action $\mathbf{G} \times F \rightarrow F$ defined by:

$$(\lambda, f) \mapsto f' = \lceil \lambda f \rceil \quad (\lambda \in \mathbf{G}). \quad (12.12)$$

(This formula is not strictly an action, because the ‘ceiling’ function does not preserve addition. However, it can be viewed as an ‘approximate action’ and the resemblance justifies the terminology in the present analogy.) The bundle resulting from this structure semigroup \mathbf{G} (analogous to a structure group but lacking inverses) is depicted in Fig. 12.5b, c right where we plot the genes in the base forming a ring to which we attached a space \mathbb{Z}_+ denoting the number of genes f in each fiber which can be changed by the action of \mathbf{G} .

The connection associated with edge $x \rightarrow y$ in the base of this graph is a map $g_{x \rightarrow y}$ of the fiber F_x associated with gene x of the base to the fiber F_y above the gene y that defines

the parallel transport of an element f of the fiber along the edge. That is, the expression $g_{x \rightarrow y} \circ f \in F_y$ is the result of the parallel transport of $f \in F_x$ to $y \in B$.

The gauge transformation in the fiber is induced by the function of the base $\lambda(x) : B \rightarrow \mathbf{G}$ defined by: $F_x \rightarrow [\lambda(x)F_x]$, for any $x \in B$. That is, this is a gauge transformation that changes the number of genes in the fiber.

While the action of \mathbf{G} over the fibers changes the number of genes in the fiber, it is also required to leave the dynamics invariant, since the expression activity in the base does not change. Thus, the redundant creation or deletion of a gene in the fiber needs to be accompanied by a rescaling in the interaction with the input genes to the fiber (Golubitsky and Stewart, 2023, Section 8.8). This is analogous to the rescaling of the exchange rates in the financial model when the currency is changed arbitrarily. In the genetic model, the addition of a node in, for instance, fiber F_y creates a new regulatory edge from gene x to the newly created gene in fiber F_y . This change in the interaction is reflected by the action of the group \mathbf{G} on the connection. The change in the connection is necessary to keep the dynamics in the graph invariant under the arbitrary increase/decrease in the fiber.

Genetic connection

We define a genetic connection, which provides the rule of parallel transport of an element f in a fiber by analogy with the financial fiber bundle. We associate an independent element of \mathbf{G} with each directed edge in the graph connecting a pair of genes in the base. The connection between gene x and gene y in the base is an operator $g_{x \rightarrow y} \in \mathbf{G}$ that belongs to the structure group \mathbf{G} and acts from a fiber F_x to a neighboring fiber F_y along the base:

$$g_{x \rightarrow y} : F_x \rightarrow F_y. \quad (12.13)$$

This connection measures the necessary adjustment in the number of interactions of the gene at base x due to a gauge change in the number of genes at base y .

Using $g_{x \rightarrow y}$ we can define parallel transport of an element f of the fiber F_x along the path $x \rightarrow y$ by $g_{x \rightarrow y}f \in F_y$. The result of a parallel transport of an element of a fiber along two paths with the same initial and end points can be different. The curvature of the graph quantifies this difference which is a measurement of the 'spiritus' or 'force' in the system.

We recall that, in the financial model, the parallel transport operator on the edge $x \rightarrow y$ connecting two countries in the base is a factor by which we convert an asset in the currency of x to the currency in y . This is the factor by which a given currency needs to be multiplied as a result of the operation represented by the edge from $x \rightarrow y$ in the base. Analogously, the parallel transport operator in the genetic base from gene x to gene y is the number of genes that a single gene x in the base regulates in the fiber at position y in the base. For instance, in the genetic network in Fig. 12.5b we have the connections:

$$g_{x \rightarrow y} = 3, \quad g_{y \rightarrow z} = 2, \quad g_{z \rightarrow x} = 1, \quad (12.14)$$

since a base gene at x regulates 3 genes in the fiber F_y at y , a base gene at y regulates 2 genes in F_z and the base at z regulates one gene in F_x .

The operator of parallel transport along the cycle from $x \rightarrow x$ in Fig. 12.5b is defined as

the product of the operators of parallel transport along the edges that constitute the cycle (Ilinski, 2001):

$$g_{x \rightarrow x} = g_{x \rightarrow y} \times g_{y \rightarrow z} \times g_{z \rightarrow x}. \quad (12.15)$$

A parallel transport of one unit of expression starting at gene x will result into 3 units at y and then $6 = 3 \times 2$ units at base y and back to 6 units at base x . The ratio between the initial unit of expression and the final after traversing the cycle in the graph is a measurement of the curvature tensor \mathcal{R} of the fiber bundle, which is defined as (Ilinski, 2001):

$$\mathcal{R} = g_{x \rightarrow y} \times g_{y \rightarrow z} \times g_{z \rightarrow x} - 1. \quad (12.16)$$

Curvature

In this particular case, the curvature of the bundle in Fig. 12.5b is $\mathcal{R} = 5$, a positive curvature that indicates the existence of a ‘genetic force’ or interaction. This positive curvature makes the messenger TFs go around the cycle in an analogous way to how arbitrage triggers speculators to move currency between countries in a speculative circuit; it is also analogous to electrons moving in cycles under the influence of a non-zero magnetic flux. All of these examples can be captured by the concept of non-zero curvature in the fiber bundle (Maldacena, 2015). This definition of curvature is motivated by the similar definition for a lattice (Ilinski, 2001), and gives the correct curvature for electromagnetism in the continuum limit when the lattice size shrinks to zero.

We see that the curvature of the circuit of Fig. 12.5b is positive. In the language of financial models, starting at x , the speculators (TFs) in the circuit can go around the financial circuit (\mathbb{Z}_3 ring) and take advantage of the ‘arbitrage opportunities’ (connections) in exchange rates, and get richer (gain in genetic activity) when they return to x . Thus a non-zero positive curvature measures the excess return (increase in expression activity) of the financial operation (activation of the genetic circuit).

Zero or negative curvature in a genetic circuit requires the existence of repression interactions. A repressor edge gives rise to a connection that is smaller than 1. It is analogous to the exchange rate from Argentina to USA in the financial model of Fig. 12.3a with $R_{\text{pesos} \rightarrow \text{USD}} = 500\text{pesos}/1\text{USD}$, since the dollar is stronger than the peso; and also to the exchange rate from USA to UK with $R_{\text{USD} \rightarrow \text{£}} = 1\text{£}/2\text{USD}$, since the pound is stronger than the dollar.

A circuit with repression is shown in Fig. 12.5c, where the edge from z to x is now a repressor of activity of gene x . The connections in this case are:

$$g_{x \rightarrow y} = 2, \quad g_{y \rightarrow z} = 1, \quad g_{z \rightarrow x} = 1/2, \quad (12.17)$$

where the repressor $z \rightarrow x$ is represented by the connection $g_{z \rightarrow x} = 1/2$, which is smaller than 1. Going around the circuit, the parallel transport operator is $g_{x \rightarrow x} = 1$ and the curvature is $\mathcal{R} = 0$ indicating that there is no excess gain of expression activity (or arbitrage in the financial model). Larger repression strength in $g(z \rightarrow x)$ would lead to smaller connection and negative curvature, analogous to a loss of money for the speculators in the financial circuit.

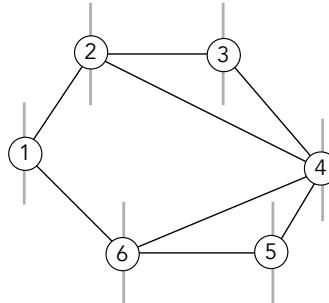


Fig. 12.7

Elementary cycles. The cycle 2342 is elementary while 124561 is not, since here we can consider the edge between 4 and 6 to obtain a pair of cycles 12461 and 4564. The operator of parallel transport along 124561 is the product of the operators along the elementary cycles 12461 and 4564.

Generalization to cycles of any length

The framework can be generalized to discrete cycles of length n in the base defined as $\Gamma = \{x_1, x_2, \dots, x_n, x_1\}$, where the gene x_1 is the initial and end point of the cycle, and the points in the cycle are ordered (Ilinski, 2001). The connection along the cycle Γ is defined by the products of the structure group elements corresponding to each of the connections at the edges in the path $g_{x_i \rightarrow x_{i+1}}$:

$$g_\Gamma = g_{x_1 \rightarrow x_2} g_{x_2 \rightarrow x_3} \cdots g_{x_{n-1} \rightarrow x_n} g_{x_n \rightarrow x_1}. \quad (12.18)$$

The graph curvature is obtained by parallel transport around the cycle Γ in the graph with a starting point x_1 , and it is a generalization of (12.16):

$$\mathcal{R}_\Gamma = g_\Gamma - 1. \quad (12.19)$$

When there are multiple cycles in the network, the curvature is restricted to the elementary cycles in the base, as in Fig. 12.7. Elementary cycles are cycles that cannot be reduced to a pair of other cycles by the consideration of an existing link in the base; see (Ilinski, 2001) and Fig. 12.7. The curvature is then:

$$\mathcal{R}_{\Gamma_{\text{elem}}} = g_{\Gamma_{\text{elem}}} - 1, \quad (12.20)$$

where Γ_{elem} is the set of all elementary cycles in the base.

The gauge transformation in the fiber: $f_y \rightarrow \lambda(y)f_y = f'_y$ with $\lambda(y) \in \mathbf{G}$ transforms the connection as: $g_{x \rightarrow y} \rightarrow [\lambda(y)g_{x \rightarrow y}] = g'_{x \rightarrow y}$. Additionally, we can define the cross-section as a map from the base to the total graph, $\psi : B \rightarrow E$ such that $\psi(x)$ is an element of F_x for each x . The cross-section defines the covariant difference between two connected points in the base as:

$$\Delta\psi = \psi(y) - g_{x \rightarrow y}\psi(x). \quad (12.21)$$

When \mathbf{G} is a Lie group, this covariant difference transforms into the covariant derivative. By definition, when the value $\psi(y)$ is the solution of the problem of parallel transport of

an element $f = \psi(x)$ along the edge $x \rightarrow y$, then the covariant difference of ψ is zero; in symbols, $\Delta\psi = 0$.

At a given point in time, a given cross-section is realized in the network. For instance, in the graph of Fig. 12.5b the cross sections are $\psi_x = 1$, $\psi_y = 3$, $\psi_z = 2$, which represent the biological realization of the redundant genes in the fibers at a given time in evolution. This cross-sections give rise to a curvature. Evolutionary changes can occur by deletion or additions of genes to the fibers according to the gauge transformation. At a given time, a given cross-section is realized. The space of all cross-sections is the space of all possible scenarios under biological evolution, and corresponds to all possible trajectories in the language of Feynman's path integral in quantum mechanics (Ilinski, 2001; Creutz, 1983; Moriyasu, 1983).

12.7 Curvature and connection in the *E. coli* genetic network

An example of such a geometric model of a genetic network is realized in *E. coli*. This network will be studied in detail in Part II. Here we consider a sample of the strongly connected component (SCC) of the genetic network that controls the pH response of the bacterium, through maintenance of pH homeostasis and control of the principal acid resistance system, as an example of genetic fibration with geometric curvature. We consider a part of the SCC in the acid response regulation circuit shown in Fig. 12.8a. The SCC contains genes that are connected by a path between any two pair of genes in the component. This path may consist of activators or repressors, indistinctly. In other words, for every gene in the SCC, there is at least a cycle that starts and ends at that gene. An input tree for a gene in the SCC is, by definition, infinite, unless the SCC contains just one gene with no autoregulation.

The circuit consists of two fibers F_{rscB} and F_{evgA} associated with the genes *rscB* and *evgA*, respectively. These fibers are studied in detail in Chapter 15; they are examples of the star fiber and FFF.

The full SCC of the pH response has a total of 53 elementary cycles. For the sake of simplicity, to show the concept of curvature in a simplified real setting, we consider three such cycles in Fig. 12.8a. These cycles run through both fibers and pass through other genes in the pH network, starting at the master regulator *hns*, and then going to genes identified with the acid response system such as *gadX*, *gadW*, and *gadE*.

While at least a gene in the fiber must belong to the SCC, not all the genes in the fiber do. For instance, F_{rscB} consists of 10 genes, but only *rscB* belongs to the SCC. The rest of the genes in the fiber from *adcD* to *yjjP* receive only a repressor edge from *hns*, but have no outgoing edge, and therefore do not belong to the SCC. The same situation is observed at F_{evgA} ; only *evgA* sends information back to the SCC while the gene *nhaR* belongs to the fiber but not to the SCC. Thus, only one gene in each of these fibers sends back information to the SCC, and that is the only gene that belongs to the SCC. We call these two fibers by the name of that gene.

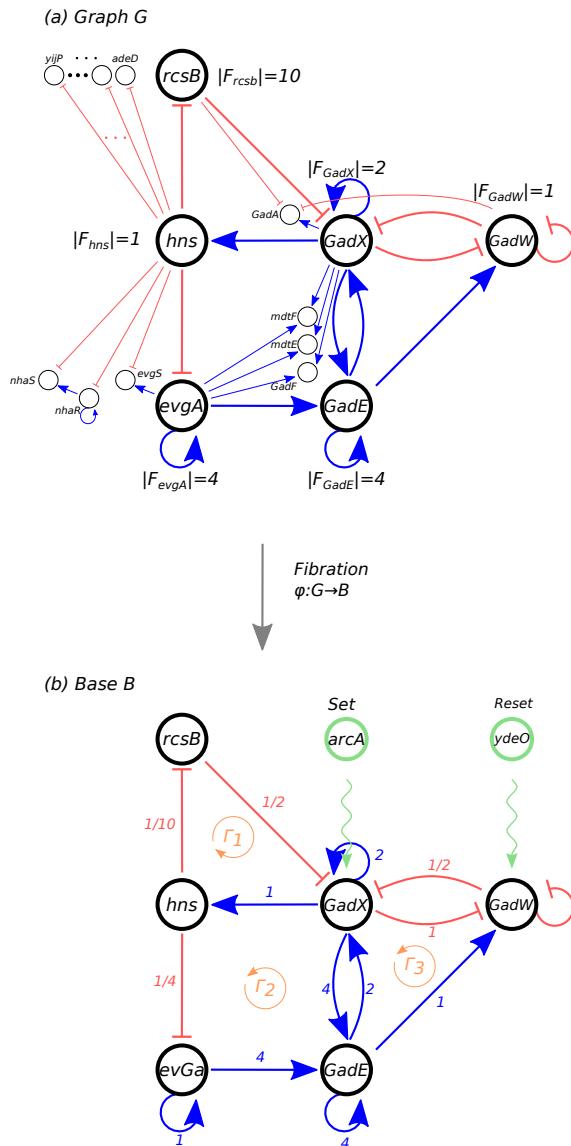


Fig. 12.8

Curvature in a genetic fibration occurring in *E. coli*. (a) Graph *G*. (b) Base *B*.

The fibration collapses the network into the base shown in Fig. 12.8b, where all genes in the base belong to the SCC. We observe three elementary cycles of path length larger than 2, namely $\Gamma_1, \Gamma_2, \Gamma_3$. For simplicity, we do not analyze the feedback loops of path length 2, nor the auto-regulation loop of path length 1. Along the edges, we assign the connections. In order to define the connections properly it is necessary to count all the genes associated with a given fiber, including the genes inside an operon. An *operon* is a set of contiguous genes that are transcribed in a row by the same RNA polymerase; they are trivially synchronized

and they form a trivial fibration. In all of this analysis we consider the genes inside operons to be a single unit. For example, the genes *evgA* and *evgS* are considered as a single unit in the operon *evgAS* (denoted *evgA* in the figure).

However, to assign the connections we need to take into account the multiplicity of the operon. In this case, the operon *evgAS* has to be counted as two genes, in order to assign the connection properly. Therefore, in the network of Fig. 12.8a, genes in an operon are considered separate for the purpose of counting the connections. However, when we count fibers (Chapter 15) we do not count operons as fibers since they are trivial. In some studies, *gadAXW* is considered to be a whole operon, but this cannot be so, since *gadY* sits between *gadW* and *gadX* and these genes have their own promoters and regulate different sets of genes. Thus we consider *gadW* to be separated from the operon *gadXA*, which is a true operon.

An example of a connection is the edge *hns*→*rscB* with a connection $g(hns \rightarrow rscB) = 1/10$ since *hns* is a repressor gene and $|F_{rscB}| = 10$. The activator edge *evgA*→*gadE* has a connection $g(evgA \rightarrow gadE) = 4$ since $|F_{gadE}| = 4$. The four genes of this fiber are *gadE*, *gadF* and *mdtE*, *mdtF*, which form an operon, but they are considered separate only for the purpose of defining the connection, as discussed above. We also assign connections to the self-loop as indicated in the figure.

When we consider the elementary cycle $\Gamma_1 = \{hns \rightarrow rscB \rightarrow gadX \rightarrow hns\}$ (excluding the self-loop at *gadX* in Fig. 12.8b) we obtain a negative curvature:

$$\mathcal{R}_{\Gamma_1} = \frac{1}{10} \times \frac{1}{2} \times 1 - 1 = -0.95 \quad (12.22)$$

indicating a loss of expression activity along the cycle. The curvature along $\Gamma_2 = \{hns \rightarrow evgA \rightarrow gadE \rightarrow gadX \rightarrow hns\}$ is positive:

$$\mathcal{R}_{\Gamma_2} = \frac{1}{4} \times 4 \times 2 \times 1 - 1 = 1, \quad (12.23)$$

indicating a gain of expression activity along this cycle. Finally, the cycle $\Gamma_3 = \{gadX \rightarrow gadE \rightarrow gadW \rightarrow gadX\}$ has a positive curvature too:

$$\mathcal{R}_{\Gamma_3} = 4 \times 1 \times \frac{1}{2} - 1 = 1. \quad (12.24)$$

This last circuit is interesting since it adds another form of regulation to the activity along the cycles. The genes *gadX* and *gadW* form a toggle-switch, like the circuit shown in Fig. 8.4, through their repressor feedback loop. This configuration is analogous to an electronic flip-flop, a device used in computers to store a bit of memory. This circuit is ubiquitous in *E. coli* and also in eukaryotes, and it has been well-studied in synthetic circuits (Gardner et al., 2000). It will be investigated in detail in Chapter 18. This flip-flop can store one bit of information by the *set* gene, in this case by the activation of gene *arcA* (see Fig. 12.8b). Gene *arcA* can turn the toggle switch on by storing a bit in *gadX* and turning off *gadW*, thus storing a memory in the activation of *gadX*. This state will remain even when *arcA* is turned off. Thus, the toggle switch works as a memory device storing one bit of information even when we turn off *arcA*. By this activation, the cycles Γ_1 and Γ_2 are activated and their negative and positive curvature would lead to repression and activation circuits, respectively. At this

point, the cycle Γ_3 will not be activated, since the toggle switch stored a 0 at gene *gadW*. This state will last even if *arcA* is switched off.

Only when the *reset* gene *ydeO* is on, is the flip-flop reset. Now *gadX* is off and *gadW* is on. At this point, the cycles Γ_1 and Γ_2 cease to operate, and they remain so even if the reset gene *ydeO* is turned off again. Thus, again, the flip-flop device remembers the state. Only when the system is reset by *arcA* will the Γ_1 and Γ_2 cycles be activated again, and the process can be repeated between the set and reset states.

These circuits represent examples of how fibrations, curvature and memory determine the computational logic underlying the function of the regulatory network. Altogether, the symmetries of the fibers, the geometry of the cycles, and the memory devices turning the cycles in the network on and off, contribute to the computational logic of the gene regulatory code. These structures will be studied in detail in Part II of this book. We will show that they characterize the structure of biological networks and contribute to the phenotype by determining the functionality of genes, proteins and metabolites in the cell.

Software and Algorithms to Perform Fibration Analysis

In the previous chapters, we have seen that fibers of fibrations predict synchronization in networks. However, we have not yet discussed how such symmetries can actually be found. This chapter answers these questions in two ways. On the one hand, we consider the problem from an algorithmic viewpoint: How difficult is it to discover symmetries? Is it easier or harder to find fibration symmetries as opposed to automorphism symmetries? What methods do we currently know to find them? And what do these questions mean exactly? On the other hand, we provide a list of pointers to actual software tools that are typically used in the context of bioinformatics and dynamical systems to find symmetries. Readers who are not interested in delving into the deeper algorithmic aspects can safely skip to Section 13.7.

13.1 Finding symmetries

To summarize in a nutshell what we have seen so far, biological structures and dynamical systems can be seen as networks of objects showing various degrees of symmetry, which in turn determine synchronization in the behavior of the objects composing the network. The types of symmetry observed in such systems depend on the deep nature of what we are looking at: classical physical systems tend to display a more rigid form of symmetry, captured by the group of automorphisms of the network; biological systems tend to display symmetries that are better described by fibrations. This chapter explains the algorithms to find these symmetries.

The algorithms we describe take as input a graph, and produce as output a partition of its nodes. The questions outlined above can therefore be rephrased more precisely as follows: if we have a graph, how can we find the node partition induced by a certain kind of symmetry? This is an algorithmic question that implies, also, a question of complexity: how difficult is it to find symmetries? We discuss both algorithms and complexity in this chapter, focusing on fibrations (in particular, symmetry fibrations). We start with automorphisms: a pleasant and important fact is that finding automorphism symmetries is apparently harder than finding fibration symmetries.

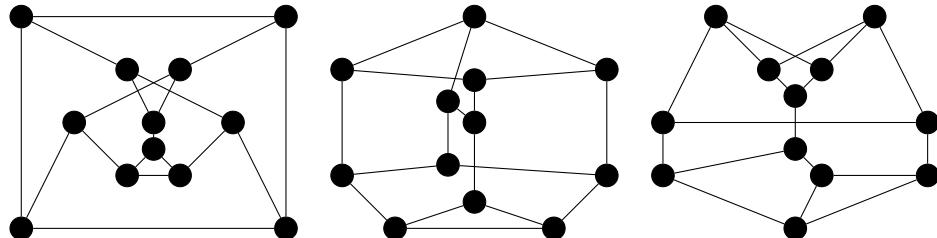


Fig. 13.1

Graph isomorphism problem. Are any of these three undirected graphs isomorphic?

13.2 Finding the automorphisms of a graph

Before embarking on a search for fibration symmetries, we discuss the more classical problem of finding automorphisms. How difficult is it to find the automorphisms of a graph? Recall that an automorphism of a graph G is a bijective graph homomorphism $\varphi : G \rightarrow G$; i.e., a function (in fact, a pair of functions, one for the vertices and one for the nodes, though for graphs with no parallel edges the former uniquely determines the latter). This search turns out to be equivalent from a computational viewpoint to another, which is subtly different but has the same answer: how difficult is it to decide whether two graphs are isomorphic? Given two graphs G and H , does there exist an isomorphism (i.e., a bijective homomorphism) $\varphi : G \rightarrow H$?

This is the *graph isomorphism problem*, and it is one of the most famous problems in computer science. Apart from its many applications, it is also interesting from a theoretical viewpoint, as it is one of the few real-world problems that are currently not known to be either polynomial-time solvable or NP-hard (i.e., unlikely to be solvable in polynomial time), see (Babai, 1996, 2016).

To give an idea of why this problem is difficult, consider the three graphs in Fig. 13.1. Can the reader tell which of these graphs are isomorphic, if any?

We might object that drawings are not how graphs are normally provided as input to algorithms, which is true. A more standard way would be to specify them as a list of edges, or using adjacency matrices. Fig. 13.2 shows three adjacency matrices, and Fig. 13.3 shows the same graphs but in the form of a list of edges. Are any of these graphs isomorphic?

These examples already suggest that finding whether two graphs are isomorphic (or, equivalently, looking for the automorphisms of a graph) is not an easy task. As we said above, it falls into an unusual category: no known algorithm solves it in polynomial-time, but there is no proof that it is NP-hard—a property that most computer scientists would interpret as ‘not having any polynomial-time solution’ (Garey and Johnson, 1979).

Regardless of the actual complexity of this problem, though, it is often easy to give a *negative* answer to the graph isomorphism question: for instance, if two graphs have a different number of vertices or a different number of edges, they *cannot* be isomorphic. This is an example of a (*polynomial*) *graph isomorphism test*, or GIT: it is something that we can be computed easily (‘polynomial’ here means ‘in polynomial time’) and that sometimes lets us rule out the possibility of two graphs being isomorphic.

$$\begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \quad \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \end{pmatrix}$$

Fig. 13.2

Graph isomorphism problem. Are any of these three graphs, represented by adjacency matrices, isomorphic?

0	3	0	8	0	4
1	2	1	2	0	6
1	6	1	4	0	7
2	1	1	6	1	2
2	3	2	1	1	5
2	5	2	4	1	9
2	8	2	5	2	1
2	9	2	7	3	5
3	0	2	8	4	0
3	2	3	4	4	9
3	4	3	8	5	1
4	3	3	9	5	3
4	7	4	1	5	7
4	9	4	2	6	0
5	2	4	3	6	7
5	9	4	5	6	9
6	1	4	6	7	0
6	8	5	2	7	5
6	9	5	4	7	6
7	4	6	1	7	8
8	2	6	4	7	9
8	6	6	7	8	7
8	9	7	2	8	9
9	2	7	6	9	1
9	4	8	0	9	4
9	5	8	2	9	6
9	6	8	3	9	7
9	8	9	3	9	8

Fig. 13.3

Graph isomorphism problem. Are these three graphs, represented as lists of edges, isomorphic?

There is a number of such tests, and each test, if it succeeds, provides a negative answer to whether the two graphs are isomorphic. However, if all such tests succeed, whether the graphs are isomorphic remains unknown. Although this conundrum is still unsolved, there are algorithms that are very efficient in practice, and that can solve the problem for many examples of interest. These algorithms do not work in polynomial time, so we should expect them to run fast typically only on small graphs, but to become slower and slower as

the graph size increases. The most famous of these algorithms is the Nauty algorithm of Brendan McKay (McKay, 1990, 1981; McKay and Piperno, 2014).

Algorithms to find the coarsest equitable partitions, which we cover in the next section, were originally studied (starting from the late 1960s) in the context of the graph isomorphism problem, i.e., precisely as polynomial GITs. We must also mention the most famous of all graph isomorphism tests, the *Weisfeiler–Lehman test* (Weisfeiler and Leman, 1968), also called the WL-test. The WL-test is in fact a hierarchy of graph isomorphism tests depending on a dimension k , and it turns out (Huang and Villar, 2021) that the color refinement algorithm (which, we shall see, is the simplest way to find the coarsest equitable partition) is equivalent to the Weisfeiler–Lehman test of dimension 1, also called 1-WL. WL tests have been revived recently because it was discovered that they are a quite powerful tool to analyze the expressivity (i.e., computational capability) of graph neural networks (Xu et al., 2018). We discuss the WL test and graph fibrations further in Chapter 26.

13.3 Finding the fibration symmetries of a graph

Recall from Definition 5.5 that an equitable partition is a partition $\mathcal{S} = \{S_1, \dots, S_K\}$ of the set V of nodes of a network $G = (V, E)$, such that each node in cluster S_μ has the same number $k_{\nu\mu}$ of incoming edges from nodes in cluster S_ν , for all $1 \leq \mu, \nu \leq K$. In other words, equitable partitions are ‘special’ partitions of the set V of nodes of a graph G , whose structure is related to the structure of the graph G . These partitions are the balanced colorings or fibers of the network.

For obvious reasons, the singleton partition (all nodes in different clusters) is equitable, whereas the discrete partition (all nodes in the same cluster) is equitable only for some graphs (precisely: only if all nodes have the same number of incoming edges). But, in the general case, is there a *coarsest* equitable partition? We already know that the answer is positive, as we learned in Chapter 5, but the problem by itself is far from trivial. Historically, the first formal proof of the existence of a coarsest equitable partition was obtained by Cardon and Crochemore (1982), but the intuition behind that dates back to Unger (1964). While we are not interested in discussing their proof here, it is useful to recall what we already know about equitable partitions (balanced colorings, fibers), fibrations and input trees. We will get back to Cardon–Crochemore and to algorithms to find equitable partitions in subsequent sections.

From Section 6.4 we know that equitable partitions of a graph are the same as fibers of surjective fibrations. In other words, if we have a surjective fibration $\varphi : G \rightarrow B$, then the fibers of φ are an equitable partition of G ; on the other hand, if we have an equitable partition \mathcal{P} of G , there exists a graph B and a surjective fibration $\varphi : G \rightarrow B$ such that the clusters of \mathcal{P} are the fibers of φ . The base B may need to be a multigraph, even if G is just a simple graph.

So, in our quest for the coarsest equitable partition of G , i.e., the partition with the minimal number of colors (fibers), we may try to find a surjective fibration of G onto some ‘smallest’ base B . Here we are implying not only that B should have the minimal possible

number of nodes, but also that for every surjective fibration $\varphi : G \rightarrow H$ (with $H \neq B$) there should be another fibration $\psi : H \rightarrow B$. (Actually, this property holds automatically because the set of equitable partitions forms a lattice under refinement, see Section 6.10.) The latter fibration may be an isomorphism, if H is already ‘smallest’ from the viewpoint of its number of nodes; otherwise (i.e., if H has more nodes than B) ψ will actually collapse some nodes.

We can now use another result that has previously been established: if $\varphi : G \rightarrow B$ is a fibration, then for every fiber the input trees of the nodes of G in that fiber are isomorphic, and they are also all isomorphic to the input tree of their image in B . In other words, if $\varphi(x_1) = \varphi(x_2) = y$ then the input trees of x_1 and x_2 in G are isomorphic, and they are isomorphic to the input tree of y in B .

This observation suggests how to construct the minimal base \hat{G} : take the graph G and collapse all the nodes that have isomorphic input trees. Edges are defined in \hat{G} in such a way as to preserve input trees, but if we are interested only in finding the coarsest equitable partition of G , we do not need to construct the edges of \hat{G} .

Algorithm 1 Finding the coarsest equitable partition by building truncated input trees.

Input: A graph $G = (V, E)$.

Output: A coloring of the nodes representing the coarsest equitable partition of G .

- 1: For every $x \in V$ build the input tree $T(x)$ truncated at the first $|V| - 1$ levels
 - 2: Arbitrarily assign a number $v(T)$ to each of the non-isomorphic input trees T
 - 3: Let κ be defined by $\kappa(x) = v(T(x))$
-

This conceptual construction is not very useful, even though Norris’s theorem (Norris, 1995), see Section 5.7, implies that we do not need to look at the whole (typically: infinite) input tree to check for isomorphism—only the first $|V| - 1$ levels suffice. So this construction is a genuine algorithm, but a very inefficient one. It is, nonetheless, formally described in Algorithm 1.

In the following Section, we describe more efficient algorithms based on finding a minimal balanced coloring with refinement algorithms, and also discuss the algorithmic problem in historical perspective.

13.3.1 Coarsest equitable partition and tests for graph isomorphism

In Section 13.2 we saw that a polynomial graph isomorphism test, or GIT, is a polynomial-time algorithm that, given two graphs G_1 and G_2 , either outputs that they are not isomorphic ($G_1 \not\cong G_2$), or it concludes that they *may be isomorphic* (meaning that it was unable to prove the opposite). In the latter case, we must either try with some other GIT, or we can try to see if the graphs are actually isomorphic in a brute force way. Sometimes, when the GIT is unable to conclude that $G_1 \not\cong G_2$ it can provide some additional information that can be used to speed up the brute-force search. Nauty (McKay, 1990) typically applies a

certain number of GITs in order, and uses brute force only if all GITs fail to conclude that $G_1 \not\cong G_2$.

To understand the relation between GIT's and equitable partitions, suppose that we want to determine whether $G_1 \cong G_2$. Instead of looking at the two graphs separately, we look at their disjoint union $H = G_1 \cup G_2$: this is the graph obtained by placing G_1 and G_2 ‘side by side’ without adding any other edge.

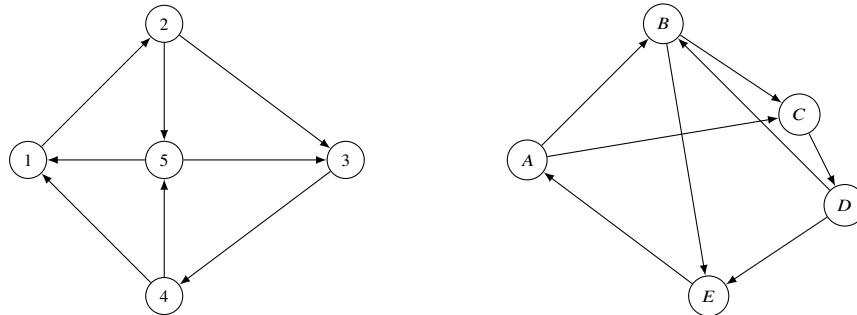


Fig. 13.4

The disjoint union of two graphs. A graph H obtained as the disjoint union of two graphs G_1 and G_2 , drawn on the left and right, respectively: a disjoint union does not add any further arc, it is just a juxtaposition of the two graphs. This example is taken from (Unger, 1964).

Take the coarsest equitable partition \mathcal{P} of H . If the two graphs are isomorphic (that is, essentially, if they are the same graph up to node identity), then every cluster of \mathcal{P} will contain the same number of nodes in G_1 and in G_2 . On the other hand, if we find that there is some cluster containing a different number of nodes from G_1 and G_2 , then for sure $G_1 \not\cong G_2$.

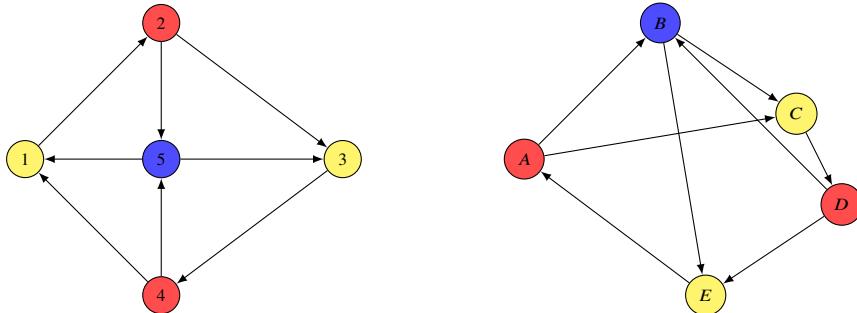


Fig. 13.5

Computing the coarsest equitable partition of the disjoint union. The coarsest equitable partition of the graph H in Fig. 13.4.

If we look at the coarsest equitable partition of the graph in Fig. 13.4 we obtain the partition shown in Fig. 13.5. This partition has three clusters (red, yellow and blue). Both G_1 and G_2 contain two yellow nodes, two red nodes and one blue node. So we can conclude that G_1 *may* be isomorphic to G_2 . So this GIT is inconclusive.

However, any isomorphism $\alpha : G_1 \rightarrow G_2$, if one exists, must respect the colors of the coarsest partition. For instance, $\alpha(5) = B$ necessarily. Intuitively, this constraint can be explained as follows: in both graphs there are only three nodes with two incoming edges: in G_1 they are 1, 3 and 5, whereas in G_2 they are B, C, E . Since $\alpha(5)$ must be a node of G_2 with two incoming edges, it must be one of B, C or E . On the other hand, the two incoming edges of 5 both come from nodes with just *one* incoming edge, whereas one of the two incoming edges of C has source B , which has two incoming edges. The same is true of E . Hence the only possibility is that $\alpha(5) = B$. This is probably better appreciated by looking at the minimal base \hat{H} (Fig. 13.6): both the yellow and the blue node have in-degree two, but only the blue node has two incoming edges both arriving from a node with in-degree one (the red node).

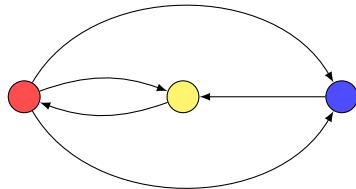


Fig. 13.6

Minimal base. The minimal base \hat{H} of the graph H in Fig. 13.5: the colors of the nodes indicate the fibers of the fibration symmetry.

So we conclude that G_1 and G_2 may be isomorphic: in other words, our GIT could not exclude the two graphs from being isomorphic. In fact, we can check (e.g., by brute force) that they are.

13.3.2 The color refinement algorithm

The first algorithm to find the coarsest equitable partition of a graph was originally proposed as a GIT by Unger (1964). The algorithm that Unger provided to find the coarsest equitable partition of a graph G is today called the *color refinement algorithm* (or ‘naive vertex classification’). It is described in Algorithm 2. Building on Unger’s method, a further improvement was proposed (always starting from the coarsest equitable partition) in (Corneil and Gotlieb, 1970).

Algorithm 2 Color refinement algorithm (Unger, 1964).**Input:** A graph $G = (V, E)$ **Output:** A coloring of the nodes representing the coarsest equitable partition of G

- 1: Let κ_0 be a coloring such that $\kappa_0(x) = 0$ for all $x \in V$
 - 2: Let $c_0 = 1$ (the number of colors used by κ_0)
 - 3: Let $i = 0$ (the number of iterations so far)
 - 4: **while** true **do**
 - 5: For each node x of V consider the multiset $M_{i+1}(x) = \{\kappa_i(s(e)) \mid e \in E, t(e) = x\}$
 - 6: Let c_{i+1} be the number of distinct multisets $M_{i+1}(x)$ (for $x \in V$)
 - 7: Arbitrarily assign a number $v(A)$ to each of the c_{i+1} multisets A
 - 8: Let κ_{i+1} be a coloring such that $\kappa_{i+1}(x) = v(M_{i+1}(x))$ for all $x \in V$
 - 9: Stop if $c_{i+1} = c_i$ (the number of colors did not change in the last iteration)
 - 10: $i = i + 1$
 - 11: **end while**
 - 12: Output κ_i
-

It is called a ‘refinement’ algorithm because it starts with the coarsest partition (the one in which all nodes have the same color), κ_0 , and then refines it until an equitable partition is obtained. Cardon and Crochemore (1982) proved that the process does eventually stop at the coarsest equitable partition.

At every step, we compute the multiset of colors $M(x)$ of the incoming edges of each node, and we assign a different color to each distinct multiset. The algorithm stops when the number of colors does not change any more. A multiset allows for multiple instances for each of its elements, for example $\{a, a, b\}$ where the element a has multiplicity 2. $M(x)$ is a multiset, because x may be the target of many edges coming from nodes of the same color (either different nodes of the same color, or even the same node, if G is itself a multigraph).

The step where the algorithm turns a multiset into numbers (arbitrarily) is not strictly necessary, but it is useful if we want to be sure that we are always using ‘numbers’ as colors, and nothing more exotic (like multisets).

The running time complexity of the color refinement Algorithm 2 is $O(|E| \cdot |V|^2)$ (Cardon and Crochemore, 1982), which is $O(|V|^3)$ for simple dense graphs, and $O(|V|^2)$ for simple sparse graphs. This is not very efficient, but it is still polynomial. This method can be applied to small graphs, but scales poorly on larger graphs.

Belykh and Hasler (2011) present a simple algorithm to find the coarsest equitable partition, based on the same general principles.

13.3.3 The Cardon–Crochemore algorithm and canonical colorings

Starting from the color refinement algorithm, Cardon and Crochemore (1982) proposed a more efficient algorithm to find the coarsest equitable partition of a graph or minimal balanced coloring. The algorithm is based on Hopcroft’s technique for minimizing finite-state automata (Hopcroft, 1971). A simpler version of the Cardon–Crochemore algorithm

was presented in (Paige and Tarjan, 1987), and later a similar algorithm with the same running time was discussed in (Junntila and Kaski, 2007). Another variant of the same ideas was proposed in (Berkholz et al., 2017), and the latter is presented in Algorithm 3.

The algorithm is very technical, and we do discuss it in detail here. Essentially, in Line 7

Algorithm 3 The algorithm of (Berkholz et al., 2017), based on the same techniques (and with the same running time as) as Cardon–Crochemore’s (Cardon and Crochemore, 1982).

Input: A graph $G = (V, E)$

Output: A coloring of the nodes representing the coarsest equitable partition of G

```

1: Let  $C_1 = V$ 
2: Let  $k = 1$  (the number of colors)
3: Let  $S_R$  be the stack containing only 1 (the only color used)
4: while  $S_R$  is not empty do
5:   Let  $r = \text{pop}(S_R)$  (the topmost color on the stack)
6:   For every  $x \in V$ , let  $d_r^-(v) = |\{e \mid t(e) = v, s(e) \in C_r\}|$ 
7:   Let  $C_S = \{c \in \{1, \dots, k\} \mid \exists x, y \in C_c, d_r^-(x) \neq d_r^-(y)\}$ 
8:   for  $s \in C_S$  (in increasing order) do
9:     Let  $m = \max_{v \in C_s} d_r^-(v)$ 
10:    For  $i = 0, \dots, m$ , let  $n_i = |\{v \in C_s \mid d_r^-(v) = i\}|$ 
11:    Let  $D = \{i = 0, \dots, m \mid n_i > 0\}$ 
12:    Let  $I = \{s\} \cup \{k + 1, \dots, k + |D| - 1\}$ 
13:    Construct a bijection  $f : D \rightarrow I$  such that  $i < j$  implies  $f(i) < f(j)$ 
14:    for  $v \in C_s$  do
15:      if  $f(d_r^-(v)) \neq s$  then
16:        Remove  $v$  from  $C_s$ 
17:        Add  $v$  to  $C_{f(d_r^-(v))}$ 
18:      end if
19:    end for
20:    if  $s \in S_R$  then
21:      for  $c \in I \setminus \{s\}$  (in increasing order) do
22:        push( $S_R, c$ )
23:      end for
24:    else
25:      Let  $b = \min\{i = 0, \dots, m \mid \forall j n_i \geq n_j\}$ 
26:      for  $c \in I$  (in increasing order) do
27:        If  $c \neq b$ , then push( $S_R, c$ )
28:      end for
29:    end if
30:    Let  $k = k + |I| - 1$ 
31:  end for
32: end while
33: Output the coloring mapping  $v$  to the index  $c$  such that  $v \in C_c$ 
```

we try to understand what are the colors (clusters) that can be broken based on r (i.e., that contain at least two nodes that have a different number of incoming edges from the color r). These clusters are analyzed in increasing index order (Line 8), and for each cluster we group the vertices that differ in the number of incoming edges from r : we leave in the original cluster only those with the smallest number of incoming edges, and we move those with more incoming edges to newly created clusters (in increasing order of number of incoming edges).

The important point, for our discussion, is that the Cardon–Crochemore algorithm has complexity $O((|E| + |V|) \log |V|)$, which is $O(|V|^2 \log |V|)$ for simple dense graphs and $O(|V| \log |V|)$ for simple sparse graph.

It was recently proved (Berkholz et al., 2017) that no algorithm can be more efficient than Cardon–Crochemore, so $O((|E| + |V|) \log |V|)$ is the best possible bound for finding the coarsest equitable partition of a graph. (However, the result of (Berkholz et al., 2017) is not for the standard general RAM computation, but only for the special class of refinement-based algorithms.)

Now Cardon–Crochemore, like all the algorithms described so far, produces an equitable partition (in fact: the coarsest equitable partition) in the form of a coloring. In other words, it takes as input a graph $G = (V, E)$, and outputs a function $\kappa : V \rightarrow C$ over some fixed set of colors C such that two nodes are in the same part of the partition if and only if they have the same color.

But this coloring is not guaranteed to be ‘canonical’. An algorithm taking as input a graph G produces a *canonical coloring* if the output coloring remains the same when the algorithm is presented with an isomorphic graph G' . In other words, if the same algorithm is provided another graph $G' = (V', E')$ that is isomorphic to G (say, via $\alpha : G \rightarrow G'$), it must output a coloring $\kappa' : V' \rightarrow C$ such that $\kappa'(\alpha(x)) = \kappa(x)$.

There are many situations where we wish to use algorithms producing canonical colorings, and Algorithm 3 does in fact produce a canonical coloring. Nonetheless, the guarantee that the coloring produced is canonical is irrelevant for our purposes, and for the discussion so far, because we never look at the actual coloring, but rather at the node partition it induces.

13.4 Algorithms to find all equitable partitions

In (Kamei and Cock, 2013a), the authors studied the problem of finding not only the coarsest equitable partition, but the *whole lattice* of possible equitable partitions. We start discussing this problem with a specific concrete example.

Consider the graph C_k (the directed k -cycle), consisting of nodes $\{0, 1, \dots, k - 1\}$ and edges $0 \rightarrow 1, 1 \rightarrow 2, \dots, k - 2 \rightarrow k - 1, k - 1 \rightarrow 0$; in Fig. 13.7 we show the 12-cycle C_{12} . What are the equitable partitions of this graph? An equitable partition is a partition of the set $\{0, 1, \dots, k - 1\}$: there are overall 4,213,597 partitions of a set of 12 elements (the 12-th Bell number), but not all of them are equitable.

In fact, it turns out that the equitable partitions of a k -cycle are only as many as the

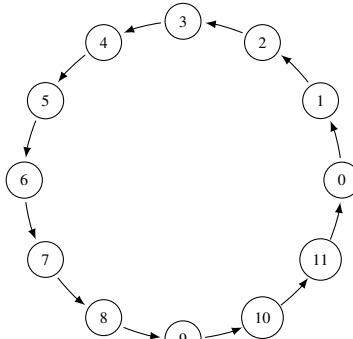


Fig. 13.7

Finding all equitable partitions of a directed cycle. The directed 12-cycle C_{12} .

number of divisors of k . More precisely, for every integer d that is a divisor of k , there is an equitable partition \mathcal{P}_d into d clusters: the partition \mathcal{P}_d puts in the same cluster all the nodes i such that $i \equiv j \pmod{d}$, that is to say, if i and j give the same remainder when divided by d .

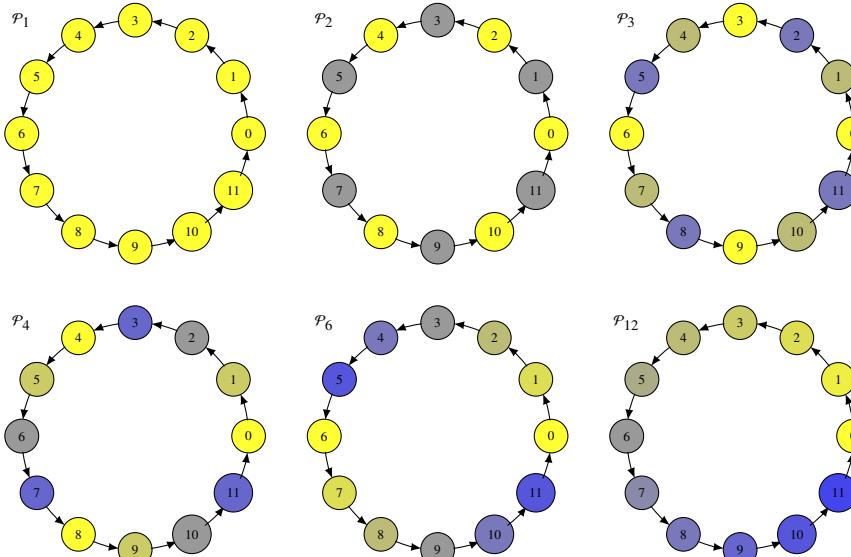
For instance, let $k = 12$; one of the divisors of 12 is $d = 2$. The equitable partition \mathcal{P}_2 contains exactly two clusters: one includes the even-numbered nodes $\{0, 2, 4, 6, 8, 10\}$, and the other contains the odd-numbered nodes $\{1, 3, 5, 7, 9, 11\}$. For each node of each of the two clusters, the number of incoming edges from the same cluster is zero, and the number of incoming edges from the other cluster is one, so this is indeed an equitable partition.

If we look at the nodes in the order of the cycle, we see that \mathcal{P}_2 alternates between two colors. There is one such partition for every divisor of 12, so we have 6 equitable partitions (one for each divisor of 12, that is, 1, 2, 3, 4, 6 and 12). The equitable partitions, represented as colorings, are shown in Fig. 13.8.

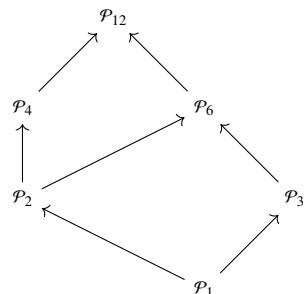
It is possible to check that these are the *only* possible equitable partitions of C_{12} , and they are all orbital. (See for example (Golubitsky and Stewart, 2023, Chapter 26), which in particular classifies all equitable partitions for uni- and bi-directional rings with nearest-neighbor connections.) If we look at them from the viewpoint of refinement, we obtain the scenario depicted in Fig. 13.9, called the *Hasse diagram* or *lattice* of equitable partitions under refinement. The elements shown in the diagram are the equitable partitions of C_{12} , and they are represented in such a way that it is possible to understand which relations are finer and which are coarser. Finer relations appear above, coarser relations below, and arrows determine which relations are refinements of which. As expected, there is a *finest* equitable partition \mathcal{P}_{12} : this is the finest possible partition, the singleton partition that puts all nodes into different clusters. But there is also a coarsest equitable partition: in this example, it is the partition \mathcal{P}_1 which is actually the trivial partition, the coarsest of all.

In Fig. 13.10 we look at some of the equitable partitions of C_{12} as fibrations over some graph. All the base graphs, in this case, are themselves cycles. More precisely, the equitable partition \mathcal{P}_d is the fiber of a fibration $\varphi_d : C_{12} \rightarrow C_d$, which maps node i to node $i \pmod{d}$.

Both (Kamei and Cock, 2013a) and (Aguiar and Dias, 2014) provide a general algorithm to determine the full lattice of all equitable partitions, as we did for the example of C_{12} .

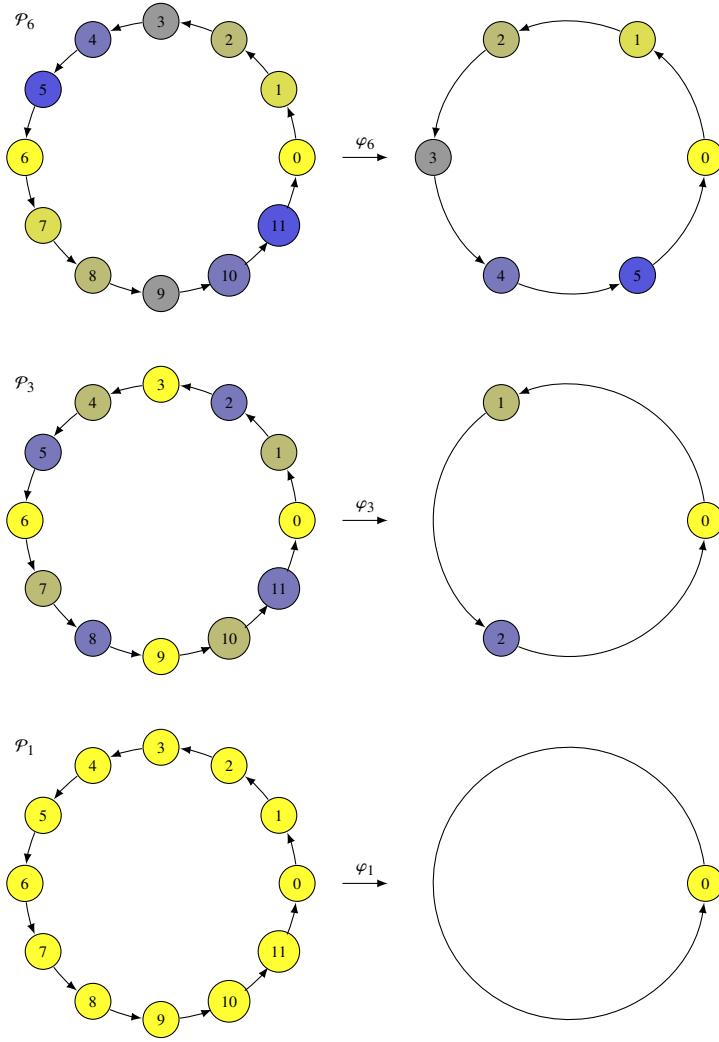
**Fig. 13.8**

The equitable partitions of C_{12} . The partition \mathcal{P}_d puts in the same cluster two nodes i and j such that $i \equiv j \pmod d$. For instance, \mathcal{P}_2 has two clusters: one for the odd-numbered nodes and one for the even-numbered nodes.

**Fig. 13.9**

The equitable partitions of C_{12} , ordered by refinement. A partition \mathcal{P} is finer than \mathcal{P}' if there is a directed path of arrows from \mathcal{P}' to \mathcal{P} . For instance \mathcal{P}_1 is coarser than all the other partitions; \mathcal{P}_2 is coarser than \mathcal{P}_4 , \mathcal{P}_6 and \mathcal{P}_{12} , finer than \mathcal{P}_1 but incomparable with (i.e., neither coarser nor finer than) \mathcal{P}_3 .

While they prove that the problem is intractable in general, their algorithm can deal with networks of up to 15 nodes—even more if the synchrony patterns are large.

**Fig. 13.10**

Fibrations of C_{12} . Some of the equitable partitions of C_{12} , shown as fibers of surjective fibrations over some base graph (on the right). The fibration acts on the nodes as suggested by colors, whereas the fibration on edges is the only possible one.

13.5 How to extend fibration algorithms to take noise into consideration

The presence of imperfections in real data is a major hurdle in biological network analysis: not all the systems in biology present the perfect symmetries that we have discussed so far. Indeed, biological data are inherently noisy and disordered, and it would be naive and unrealistic if we were to expect that genomes and connectomes would give us the perfect symmetries discussed theoretically. Analogously, biological systems will display approximate cluster synchronization, as opposed to exact cluster synchronization, something that was first pointed out in (Sorrentino and Pecora, 2016).

13.5.1 Pseudosymmetries

This observation motivates the development of a theory of approximate symmetries. The general idea is to take a notion of symmetry (e.g., automorphism symmetry or fibration symmetry) and to relax the conditions it imposes to allow for a limited number of *exceptions*. A first step in this direction was proposed in (Morone and Makse, 2019), with the concept of an ε -pseudosymmetry; they also observe that any ε -pseudosymmetry can be converted into a genuine symmetry of a ‘repaired’ graph.

We include this development for completeness; however, we do not recommend it as the best framework for dealing with noisy environments. First, group symmetries are not particularly useful in biology. Second, fibrations are more general and encompass group symmetries as well, making the theory of quasi-fibrations and pseudo-balanced coloring more applicable. These topics will be discussed in the following two sections.

Even the pseudo-balanced colorings addressed in Section 13.5.3 and the associated repair algorithm for restoring symmetry may not be the most effective approach. This limitation arises because the algorithm does not account for the most common scenario, which involves assisting network repair with some form of dynamical data, such as information about cluster synchronization within the system. This more useful case will be explored in Chapter 22, along with its application to brain network inference in subsequent chapters.

Recall that a permutation matrix is a matrix of zeros and ones containing exactly one 1 in every row and column. If A is the adjacency matrix of a graph G , then an automorphism of G is a permutation matrix P such that $PAP^{-1} = A$. (The matrix PAP^{-1} permutes both the rows and columns of A according to P , so this equation states that the permuted adjacency matrix is equal to the original one. This is the property required for an automorphism.)

Equivalently $PA = AP$, which we rewrite as

$$[A, P] = \mathbf{0} \tag{13.1}$$

where $[A, P] = AP - PA$ and $\mathbf{0}$ is the matrix whose entries are all zeros.

A natural extension to allow for errors is the following. Define the L^1 norm of a matrix $M = (m_{ij})$ to be

$$\|M\| = \sum_{i,j} |m_{ij}|$$

This norm has several useful properties.

- (a) If M has integer entries, as is the case for an adjacency matrix, then $\|M\|$ is a nonnegative integer.
- (b) If the entries of M are always 0 or 1, as is also the case for an adjacency matrix, then $\|M\|$ is the number of nonzero entries of M .
- (c) Therefore if M, N are two matrices, whose entries are always 0 or 1, the quantity $\|M - N\|$ is equal to the number of entries at which M and N differ.
- (d) If Q, R are permutation matrices then $\|QMR\| = \|M\|$. This holds because M and QMR have the same set of entries: the permutation matrices only permute their positions.

Fix A and suppose that Q is any permutation matrix. If Q is an automorphism then, by (13.1), $[Q, A] = \mathbf{0}$, so $\|[A, Q]\| = 0$. We replace this by the (milder) condition

$$\|[A, Q]\| \leq \varepsilon. \quad (13.2)$$

By property (d) above this is equivalent to either of

$$\|AQ - QA\| \leq \varepsilon \quad \|A - QAQ^{-1}\| \leq \varepsilon. \quad (13.3)$$

The case of interest is when $\varepsilon > 0$ is small compared to the number of entries in the matrix, but initially we consider any $\varepsilon > 0$.

Definition 13.1 The matrix Q is an ε -pseudosymmetry of A if (13.2) holds, or equivalently (13.3) holds.

For a given $\varepsilon \geq 0$, consider the set

$$\mathcal{P}_\varepsilon = \{Q \mid \|[A, Q]\| \leq \varepsilon\}.$$

This is the set of all permutation matrices Q such that $[A, Q]$ has a small L^1 norm rather than being zero. These permutations are the pseudosymmetries of G .

‘Small’ here means ‘small relative to the size of the network’, because an integer-valued norm is never ‘small’ in any intuitive sense—unless it is zero. Here, the smallest nonzero value of ε is 1. However, if we measure the network size by the number n of nodes, say, then ε/n is relatively small, and we get a good approximation when ε/n is small enough. Alternatively, we might divide ε by n^2 , the number of matrix entries, or by the number of edges, or by some other measure of network complexity.

To understand better what pseudosymmetries are, consider a fixed but arbitrary pseudosymmetry $Q \in \mathcal{P}_\varepsilon$. It need not represent an automorphism of G , but by (c) its norm is precisely the number of entries at which A and QAQ^{-1} are different. Then one of them has a 0 entry where the other has a 1 entry. We can ‘repair’ the graph by making all such entries of A equal to 1. (Alternatively, to 0.) Let A_ε be the resulting adjacency matrix, bearing in mind that this matrix depends on Q as well as on ε .

Then $[A_\varepsilon, Q] = 0$, so Q is an automorphism of the repaired graph with adjacency matrix A_ε , obtained by adding or removing some edges. Call this graph G_ε , again bearing in mind that this also depends on Q . Now:

$$\varepsilon \geq \|[A, Q]\| = \|AQ - QA\| = \|A - QAQ^{-1}\|,$$

where in the last equality, we use property (d). Adding and subtracting A_ε we obtain

$$\varepsilon \geq \|A - A_\varepsilon + A_\varepsilon - QAQ^{-1}\| = \|A - A_\varepsilon + QA_\varepsilon Q^{-1} - QAQ^{-1}\|.$$

In the last equality we used the fact that $[A_\varepsilon, Q] = \mathbf{0}$, that is, $A_\varepsilon Q = QA_\varepsilon$, hence $A_\varepsilon = QA_\varepsilon Q^{-1}$. Hence

$$\varepsilon \geq \|(A - A_\varepsilon) - Q(A - A_\varepsilon)Q^{-1}\| \geq \|A - A_\varepsilon\| + \|Q(A - A_\varepsilon)Q^{-1}\| = 2\|A - A_\varepsilon\|.$$

The latter term ($\|A - A_\varepsilon\|$) is exactly the number of edges that have been added or removed from G , and the inequality shows how this number is related to ε .

In other words, for a fixed value of ε we have an upper bound on the number of edges that can be changed to convert a pseudosymmetry Q of G into a symmetry Q of the repaired graph G_ε .

Example 19 Let G be as in Fig. 13.11 (top). The adjacency matrix is

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

This graph is obtained by changing one connection in a unidirectional ring with six nodes and nearest-neighbor arrows. It can also be seen as a unidirectional ring with five nodes 2 – 6, plus an extra input node 1.

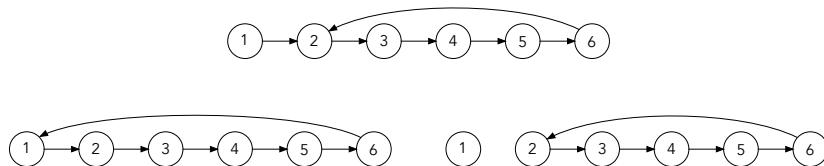


Fig. 13.11

Pseudosymmetries. *Top:* Graph with 6 nodes, exemplifying pseudosymmetries. *Bottom:* Two potential repairs. *Left:* Symmetry group \mathbb{Z}_6 . *Right:* Symmetry group \mathbb{Z}_5 .

Let Q be the matrix of the 6-cycle (123456), so that

$$Q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Then

$$[A, Q] = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & -1 \end{bmatrix}$$

and $\|[A, Q]\| = 4$. Therefore Q is an ε -pseudosymmetry with $\varepsilon = 4$. The maximum value that the norm can take is 36, so this gives a proportion 1/9 of differing entries.

Alternatively, we might try the 5-cycle (23456). Now

$$Q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

so that

$$[A, Q] = \begin{bmatrix} 0 & -1 & 0 & 0 & 0 & 1 \\ 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

and $\|[A, Q]\| = 6$. Therefore Q is an ε -pseudosymmetry with $\varepsilon = 6$, which is a bigger error than 4.

In the first case the natural repaired network is a unidirectional ring on all six nodes, Fig. 13.11 (bottom left). In the second case, it is a unidirectional ring on nodes 2 – 6, with 1 as an isolated node, Fig. 13.11 (bottom right).

13.5.2 Quasifibrations

A related issue is that of statistical noise in observations. When moving to the world of fibrations, we can approach the problem of considering noise in a number of ways. One possibility, described in (Leifer et al., 2022), is to define a relaxed notion of equitable partition that explicitly takes statistical noise into account. In terms of the adjacency matrix A of a graph G , a node partition \mathcal{S} is equitable if and only if, for all clusters C, D of \mathcal{S} and for all pairs of nodes $p, q \in C$, the number of edges entering in p and q and coming from D are the same, that is

$$\sum_{j \in D, (j,p) \in E} A_{jp} = \sum_{j \in D, (j,q) \in E} A_{jq}.$$

If we let Φ denote the perturbation matrix representing the noise, we can require that

$$\sum_{j \in D, (j,p) \in E} (A_{jp} + \Phi_{jp}) = \sum_{j \in D, (j,q) \in E} (A_{jq} + \Phi_{jq}).$$

In this case \mathcal{S} is called *quasi-equitable*, up to the noise matrix Φ . Besides Φ (which contains only information on how *existing* edges should be modified), the full formulation also allows extra edges to be added, which requires a second noise matrix Ω : this case is presented in the next subsection.

Here there are two factors to play with: the number of clusters $|\mathcal{S}|$ (in particular, the number of clusters of the coarsest equitable partition) and the norm of the perturbation matrix $\|\Phi\|$. Of course, we expect these two values to be negatively correlated: if we allow for more noise, we can get a more symmetrical structure, reducing the number of clusters. Leifer et al. (2022) prove that for a fixed value of $|\mathcal{S}|$, minimizing $\|\Phi\|$ is NP-hard. However, they formulate an integer programming optimization which can be used as a heuristic to find an approximate solution for the coarsest quasi-equitable partition.

An alternative approach to noisy fibration symmetries is described in (Boldi et al., 2022). Recall that a fibration $\varphi : G \rightarrow B$ is a homomorphism that satisfies the lifting property: in other words, for every edge a of B and every node x of G in the fiber of the target of a , there must be *exactly one* edge of G with target x that φ maps to a . In a way, though, any homomorphism $f : G \rightarrow B$ can be seen as a ‘fibration with errors’. Errors can be of two types: *deficiencies* (i.e., edges that have *no* lift at a certain target) and *excesses* (i.e., edges that have *more than one* lift at a certain target). The overall number of deficiencies and excesses can be used as a measure of how much a given homomorphism deviates from being a fibration. Such a homomorphism is called a *quasifibration*.

Suppose, for a moment, that we know what an equitable partition \mathcal{S} of G looks like. That is, that we know how the nodes of G can be divided to obtain an equitable partition. Then we can consider a homomorphisms $f : G \rightarrow B$, whose fibers are exactly the clusters of \mathcal{P} , to be a fibration with errors. The question is whether we can find, among all these homomorphisms, one that is as close as possible to being a fibration, i.e., one that is a quasifibration with the smallest number of deficiencies and excesses. This problem turns out to be polynomial-time solvable in an exact way (Boldi et al., 2022).

This result is only partially satisfactory, for at least two reasons. The first is that we need to have some idea of an equitable partition to start with; this equitable partition can be obtained using various heuristics, but it is certainly a limitation that must somehow be overcome (in a heuristic way, necessarily, for otherwise the optimization problem in (Leifer et al., 2022) would not be NP-hard). The second limitation is that the algorithm makes a very specific assumption on the metric that measures the distance from the space of fibrations; using other metrics can be difficult (and, in particular, it is proved in (Boldi et al., 2022) that the general problem for arbitrary metrics is NP-hard).

13.5.3 Pseudo-balanced colorings and repair algorithm without knowledge of balanced coloring

Leifer et al. (2022) explore an alternative approach to finding fibration symmetry of a graph in the presence of noise.

The algorithm reconstructs a network by optimally selecting the minimal number of modifications needed to achieve a fiber-symmetric network. In this algorithm, the balanced coloring is not known *a priori*, and we do not know the number of colors either. Therefore, the algorithm is initially run using a fixed number of colors, and the network is reconstructed accordingly.

Afterward, the algorithm can be executed with different numbers of colors, and the best solution should be chosen based on heuristic metrics, as discussed in (Leifer et al., 2022). A simpler and more effective algorithm is developed in Chapter 22, which assumes that we know the balanced coloring from synchronization experiments.

We do not consider unweighted multigraphs, but rather weighted simple graphs. The difference is that between each source/target pair there can be only one edge, but that edge is weighted. Although the definition of fibrations is not well suited for weighted graphs (as discussed in Section 9.4), balanced colorings can easily be extended to this case. Instead of requiring that each pair of nodes with the same color c receive the same number of edges from each color class c' , we require that the *sum of the weights* of edges coming from c' should be the same. Incidentally, if all weights are integers and the weighted simple graph is interpreted as an unweighted multigraph (with weights corresponding to multiplicities), then the two notions of balanced colorings coincide.

While the scope of (Leifer et al., 2022) is broad, next we discuss the particular approach they develop to repair a given unweighted undirected graph $G(V, E)$ with adjacency matrix A , aiming to produce a pseudobalanced coloring with k colors. This is relevant to all situations where imperfect knowledge of a given graph may be affected by missing links. The underlying assumption is that the original graph G may lack some of the true fibration symmetries and, as a result, produce balanced colorings with an exceedingly large number of trivial colors.

Leifer et al. (2022) introduce the perturbation matrix $\Omega = \{\omega_{ij} \in \{0, 1\}\}$. Then they formulate the pseudo-balanced coloring integer program (PBCIP) in the quantities ω_{ij} , which minimizes the total number of added links:

$$\min \left\{ \sum_{i,j \in V, ij \notin E} \omega_{ij} \right\}, \quad (13.4)$$

This is done in a manner that makes the repaired graph G' , with adjacency matrix $A' = A + \Omega$, have a balanced coloring with k colors. This allows only adding links that are not already present, not removing them. Solving this PBCIP finds the minimal number of edges to add to a given graph to ensure a balanced coloring for a fixed number of colors. The overall repair method presented in (Leifer et al., 2022) proceeds according to the following steps:

1. Pre-processing: compute the minimal balanced coloring for the original graph G and

identify the initial non-trivial set of colors. Let C denote the number of non-trivial colors and T the number of trivial colors in this initial coloring.

2. for all k from C to $C + T$ do the following:
 - a. Solve the (PBCIC) for k colors, with added constraints to fix the nodes that were already assigned to one of the C non-trivial colors.
 - b. Construct the resulting graph G' with the added edges dictated by (PBCIP) and color the nodes according to the minimal balanced coloring of G' .
3. Evaluate the resulting T graphs to identify the number of colors of the *best* repaired graph. In general this involves both a quantitative and a qualitative assessment, which may be based on available information about the graph structure and functions.

Then Leifer et al. (2022) proceed by considering an optimization problem that looks for some balanced coloring with K colors in a given weighted graph G , which allows for the presence of noise in the following sense:

- We can introduce some (positive or negative) noise into the weights of existing edges $\Phi = (\phi_{ij})_{ij \in E}$.
- We can add some new edges E' with some weights on them: $\Omega = (\omega_{ij})_{ij \in E'}$.

The noise on existing edges and the weights of new edges can be restricted to a certain class \mathcal{R} , i.e., $(\Phi, \Omega) \in \mathcal{R}$. Also, the search space (the set of allowed colorings) is restricted by providing a coarser coloring \mathcal{F} (in the sense that we are not allowed to mix color classes that are distinct in \mathcal{F}).

This framework yields an optimization problem whose objective function is the overall noise

$$\sum_{ij \in E} |\phi_{ij}| + \sum_{ij \in E'} |\omega_{ij}|,$$

and whose constraints are the graph G , the number of colors K , the type of noise we can introduce \mathcal{R} , and the constraint \mathcal{F} on colors. While the most general version of this optimization problem is known to be computationally hard, the authors discuss an integer formulation that is at the same time amenable to polynomial-time approximation and, in practice, provides very competitive results.

13.6 How to extend fibration algorithms to take types into consideration

As discussed in Section 9.5, our algorithms should be extended to take heterogeneity into consideration. We limit ourselves to showing in Algorithm 4 how the naive color refinement algorithm (Algorithm 2) is adapted to the heterogeneous case. Essentially:

- Every time a color is assigned to some node x , both during initialization (line 1) and

- during iterations (line 8), the color takes the node type $\nu(x)$ into account (so that nodes of different types are always assigned different colors);
- Every time we look at the input set of a node x (line 5) we consider the colors of edges connecting x to its in-neighbors.

Algorithm 4 Color refinement algorithm (Algorithm 2) modified for heterogeneous graphs.

Input: A heterogeneous graph $G = (V, E)$, with $\nu : V \rightarrow T$ and $\eta : E \rightarrow T$ being the maps specifying the types of nodes and edges

Output: A coloring of the nodes representing the coarsest equitable partition of G

- 1: Let κ_0 be a coloring such that $\kappa_0(x) = \nu(x)$ for all $x \in V$
 - 2: Let $c_0 = |\nu^{-1}(V)|$ (the number of colors used by κ_0)
 - 3: Let $i = 0$ (the number of iterations so far)
 - 4: **while** true **do**
 - 5: For each node x of V consider the multiset $M_{i+1}(x) = \{\langle \eta(e), \kappa_i(s(e)) \rangle \mid e \in E, t(e) = x\}$
 - 6: Let c_{i+1} be the number of distinct pairs $\langle \nu(x), M_{i+1}(x) \rangle$ (for $x \in V$)
 - 7: Arbitrarily assign a number $v(\langle t, A \rangle)$ to each of the c_{i+1} pairs $\langle t, A \rangle$
 - 8: Let κ_{i+1} be a coloring such that $\kappa_{i+1}(x) = v(\langle \nu(x), M_{i+1}(x) \rangle)$ for all $x \in V$
 - 9: Stop if $c_{i+1} = c_i$ (the number of colors did not change in the last iteration)
 - 10: $i = i + 1$
 - 11: **end while**
 - 12: Output κ_i
-

We exemplify the execution of Algorithm 4 on the graph of Fig. 9.10. Let us first try to find its coarsest equitable partition *in absence of typing*, for instance, using color refinement (Algorithm 2): this algorithm starts with all nodes having the same color, and then refines the coloring based on the color of in-neighborhoods. In the example of Fig. 9.10, at the beginning all nodes are assigned the same color (call it ℓ_0), and at the first iteration of the algorithm

- nodes with no inputs (1, 2 and 5) all see the empty multiset $\{\}$ of colors
- nodes 3 and 4 both see a multiset with three elements (because they have exactly three incoming edges): the multiset is $\{\ell_0, \ell_0, \ell_0\}$
- finally nodes 6 and 7 both see the multiset $\{\ell_0, \ell_0\}$.

Let ℓ_1 be the color assigned to the first multiset, ℓ_2 the color assigned to the second multiset, and ℓ_3 the color assigned to the last multiset. At the next iteration

- nodes with no inputs (1, 2 and 5) all see the empty multiset $\{\}$ of colors
- nodes 3 and 4 all see a multiset with three elements (because they have exactly three incoming edges): the multiset is $\{\ell_1, \ell_1, \ell_2\}$
- finally nodes 6 and 7 sees the multiset is $\{\ell_1, \ell_1\}$.

x	$\kappa_0(x)$	$\langle v(x), M_1(x) \rangle$	$\kappa_1(x)$	$\langle v(x), M_2(x) \rangle$	$\kappa_2(x)$
1	1	$\langle 1, \{\} \rangle$	3	$\langle 1, \{\} \rangle$	3
2	2	$\langle 2, \{\} \rangle$	4	$\langle 2, \{\} \rangle$	4
3	*	$\langle *, \{1, 2, *\} \rangle$	1	$\langle *, \{2, 3, 4\} \rangle$	1
4	*	$\langle *, \{2, 5, *\} \rangle$	2	$\langle *, \{1, 4, 5\} \rangle$	0
5	5	$\langle 5, \{\} \rangle$	5	$\langle 5, \{\} \rangle$	5
6	*	$\langle *, \{1, 2\} \rangle$	0	$\langle *, \{3, 4\} \rangle$	2
7	*	$\langle *, \{1, 2\} \rangle$	0	$\langle *, \{3, 4\} \rangle$	2
	$c_0 = 4$		$c_1 = 6$		$c_2 = 6$

Since the number of colors did not change with respect to the previous iteration, the algorithm stops and the final color (corresponding to the coarsest equitable partition, i.e., to the fibration symmetry) is the one shown in Fig. 9.11.

Now, let us add a typing on top of the graph, as discussed in Section 9.6, so as to take into consideration in a proper way the nodes that have no input. Explicitly, the type function $v(x)$ is defined by:

$$v(x) = \begin{cases} x & \text{if } x \text{ has no inputs} \\ * & \text{otherwise.} \end{cases}$$

The renumbering at each step (function v in Algorithm 4 is arbitrary; here we use lexicographic ordering of the pair $\langle v(x), M_i(x) \rangle$).

The symmetry fibration obtained is shown in Fig. 9.12. Not only do all nodes with no inputs belong to different classes (this is trivial, because they do not have the same type), but also nodes 3 and 4 are different because they receive inputs from different nodes (albeit nodes with no input themselves). Here the only nontrivial fiber is the one formed by nodes 6 and 7.

An implementation of Algorithm 4 with the preprocessing required for nodes with no input is available at <https://github.com/MakseLab/FibrationSymmetries> in the form of an R package.

13.7 Software used in this book to find minimal balanced colorings

All fibration analysis done in this book has been performed using a version of the refinement algorithm developed by Leifer (2022), see also the supplementary information of (Morone et al., 2020) and (Monteiro et al., 2022). The particular implementation of the refinement algorithm explained in this section is available in R at <https://github.com/MakseLab/FibrationSymmetries>. This algorithm takes into account nodes with no inputs from the very beginning, implicitly assigning them different types (like in Section 13.6). We explain this implementation below. The reader interested in reproducing all results presented in this book can download the codes and data to perform the fibration analysis from the repository

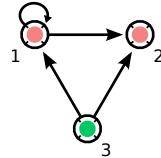
of the book at <https://github.com/MakseLab/> and <https://osf.io/4ern8/>. Appendix A lists all source code and data used in this book.

The analysis identifies fibers by searching for the minimal balanced coloring, i.e., the network's coarsest equitable partition (minimal base). The most efficient algorithm is the one developed by Paige and Tarjan (1987), and has time complexity $O(M \log N)$ and space complexity $O(M + N)$. Implementation of this algorithm, however, is fairly complicated, and in the interest of giving a concise explanation, we present our implementation of the algorithm developed in (Leifer, 2022) and (Morone et al., 2020) based on (Kamei and Cock, 2013a). This algorithm has runtime complexity $O(N^2 \log N)$ and space complexity is $O(N^2)$.

Leifer (2022) defines the *Input Set Color Vector (ISCV)*. Let $G = (H, \chi)$ be a colored graph with k colors of $H = (N, E)$ and $\chi : N \rightarrow (\chi_1, \chi_2, \dots, \chi_k)$, $k \in \mathbb{N}_{<|N|}$ be a function assigning each node from N a color. Let $\mathcal{P} = \{c_1, c_2, \dots, c_k\}$ be the partition corresponding to χ .

Definition 13.2 The *ISCV (Input Set Color Vector)* of node n is a k -dimensional vector $ISCV(n) = (ISCV_1(n), ISCV_2(n), \dots, ISCV_k(n))$ in which $ISCV_i(n) = |(N_{I(n)} \setminus \{n\}) \cap c_i|$.

That is, $ISCV(n)$ is a k -dimensional vector with i -th entry equal to the number of nodes of i -th color in the input set of n (excluding the node itself).



	ISCV(1)	ISCV(2)	ISCV(3)
ISCV ₁	1	1	0
ISCV ₂	1	1	0

Fig. 13.12

Input Set Color Vector. An example of an ISCV is found on the graph nodes with minimal balanced coloring.

A coloring is balanced and minimal when nodes of the same color have the same ISCVs and nodes of different colors have different ISCVs. Figure 13.12 shows an example of a minimal balanced coloring. The graph in Fig. 13.12 has 2 colors. Therefore, the ISCV is a 2-dimensional vector. Node 1 receives input from itself (red color) and from node 3 (green color); therefore, $ISCV(1) = (1, 1)$. Similarly, $ISCV(2) = (1, 1)$ and $ISCV(3) = (0, 0)$.

The algorithm for balanced coloring is based on the iterative procedure of partition refinement. The minimal balanced coloring is unique and can be obtained by refining the unit partition until no further refinement is possible.

1. Assign all nodes the same color. Set $d = 1$.
2. Find the ISCV of all nodes (In the first iteration, the ISCV is a 1-dimensional vector equal to the in-degree of the node).
3. Let m be the number of unique ISCVs.

4. If m is equal to d , stop.
5. Assign each of the m unique ISCVs a new color and recolor the graph accordingly. Set $d = m$.
6. Repeat steps 2-5 until the condition in step 4 is satisfied.

Figure 13.13 shows an example of the step-by-step execution of the software implementation. Both examples provided in this chapter involve unweighted networks, but the software can be applied to undirected and weighted networks as well.

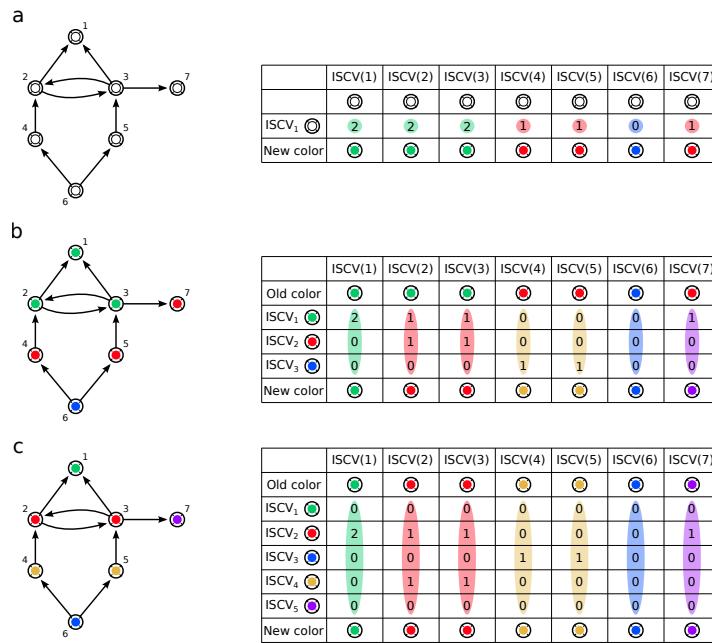


Fig. 13.13

Example of the step-by-step execution of software to find the minimal balanced coloring. (a) All nodes are assigned the color white. The ISCV corresponds to the in-degree of each node. Three unique ISCV values, (0), (1), and (2), define three new colors: green, red, and blue. (b) A new coloring is defined by three unique ISCVs in (a). The ISCV is now a 3-dimensional vector. There are five unique ISCVs, so there are five new colors. (c) A new coloring is defined by five unique ISCVs in (b). The ISCV is now a 5-dimensional vector. There are five unique values of the ISCV, so the algorithm stops.

As explained in the previous section, real applications in biology require the solution of a slightly modified problem: finding an almost minimal coloring. This modified balanced coloring algorithm essentially assigns a different color to each node that has no inputs.

The symmetry fibration obtained by this procedure is not minimal but rather ‘as minimal as possible’ given that nodes without inputs belong to different fibers. Morone et al. (2020)

offer a modification of the algorithm that obtains symmetry fibers and takes into account the biological fact that nodes without inputs are not synchronized, even though in the fibration theory, they would be put into the same fiber, since their input tree, which is a null tree, would be mathematically isomorphic. However, common sense indicates that such nodes should be put into different fibers since they do not receive the same inputs; thus, they will never synchronize.

In the analysis of real biological networks, it is rare for a situation to arise where a gene receives no inputs. If this occurs, it typically indicates that the data set is incomplete. In a regulatory network, for example, all genes must receive inputs in order to be activated or repressed. If a gene has no input, it suggests that the network is not fully complete. Regarding small molecules and metabolites, it is possible for these nodes to have no inputs when they are part of the external environment. However, in such cases, it is unrealistic to assume that all external molecules are synchronized, as the environment is constantly changing.

Thus, we modify the refinement algorithm to assume that nodes with no inputs are in different fibers, as discussed in Section 9.6. Leifer (2022) developed the implementation explained below, which takes this modification into account, producing a coloring that is not exactly minimal but differs from the minimal coloring by the number of colors assigned to the nodes with zero inputs.

This modification is a heuristic that works for all networks considered by the authors in the analysis of all data included in this book. However, in some specific examples, it may not work. We first describe the heuristic, then give some examples that can be problematic and discuss alternative approaches. The heuristic is based on two modifications applied to the algorithm:

1. Identify all nodes with no inputs from other nodes (nodes that receive only from themselves) and assign each of them a different initial color.
2. Identify all nodes with no inputs from any node (including themselves) and assign each of them its color during every iteration of the algorithm.

The first modification is aimed at dealing with the situation shown in Fig. 13.14. The ISCV of a node that receives inputs only from itself consists of one non-zero entry corresponding to the color of this node. The ISCV of any node that receives inputs only from this node is equal to the ISCV of the node, which creates fibers like those shown in blue and red in Fig. 13.14 while keeping blue and red fibers separate from each other.

The second modification deals with the more trivial situation in Fig. 13.15. The ISCVs of all nodes with no inputs are equal to the zero vector at every step of the algorithm; therefore, assigning different initial colors is not enough, and repeated assignment during every iteration is needed to maintain the partition.

Algorithm 5 Implementation of the refinement algorithm developed by Leifer (2022) to find minimal balanced colorings in graphs. This implementation is used to obtain all results displayed in this book. See also (Morone et al., 2020).

Input: A graph $G = (V, E)$ with $V = \{1, \dots, N\}$
Output: Colors of the symmetry fibers $\chi_1, \chi_2, \dots, \chi_N$, $N = |V|$

- 1: For every node j , we denote with I^j the ISCV of j , and with I_i^j its i -th component
- 2: We let $d(j, i)$ be the number of nodes of color i adjacent to node j
- 3: **if** G is directed **then**
- 4: Let z_1, \dots, z_ℓ be the nodes with no inputs
- 5: Let x_1, \dots, x_m be the nodes that receive input only from themselves
- 6: **else**
- 7: Let $\ell = m = 0$
- 8: **end if**
- 9: **for all** $j \in \{1, \dots, N\}$ **do**
- 10: Let $\chi_j = 1$
- 11: **end for**
- 12: **for all** $j \in \{1, \dots, \ell\}$ **do**
- 13: Let $\chi_{z_j} = j$
- 14: **end for**
- 15: **for all** $j \in \{1, \dots, m\}$ **do**
- 16: Let $\chi_{x_j} = \ell + j$
- 17: **end for**
- 18: **if** $\ell + m \neq N$ **then**
- 19: Let $d = \ell + m + 1$
- 20: **else**
- 21: **return** χ_1, \dots, χ_N
- 22: **end if**
- 23: **while** true **do**
- 24: **for all** $i \in \{1, \dots, d\}$ and $j \in \{1, \dots, N\}$ **do**
- 25: Let $I_i^j = d(j, i)$
- 26: **end for**
- 27: Assume that there are k distinct ISCV's, say H^1, \dots, H^k (numbering is arbitrary)
- 28: **for all** $i \in \{1, \dots, k\}$ and $j \in \{1, \dots, N\}$ **do**
- 29: **if** $I^j = H^i$ **then**
- 30: Let $\chi_j = \ell + i$
- 31: **end if**
- 32: **end for**
- 33: **for all** $j \in \{1, \dots, \ell\}$ **do**
- 34: Let $\chi_{z_j} = j$
- 35: **end for**
- 36: **if** $d = \ell + k$ **then**
- 37: **return** χ_1, \dots, χ_N
- 38: **else**
- 39: $d = \ell + k$
- 40: **end if**
- 41: **end while**

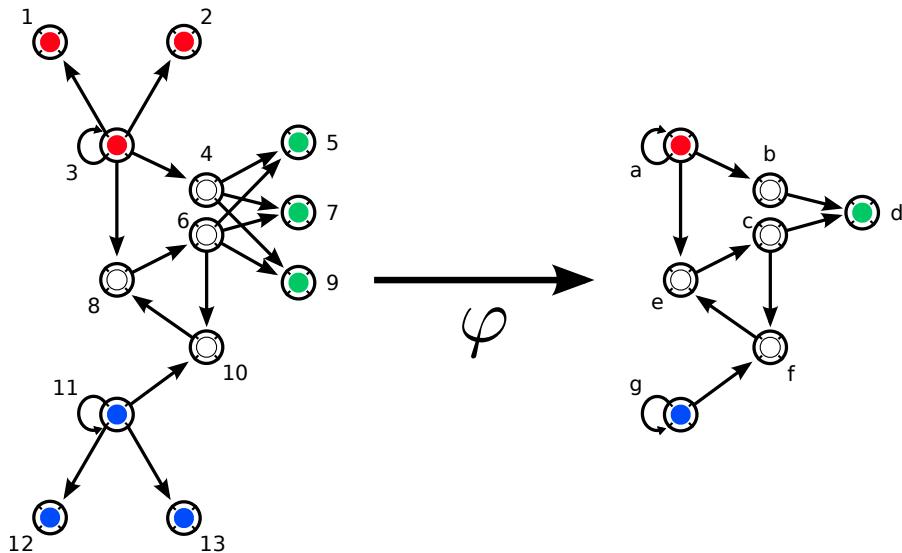


Fig. 13.14

Special cases of fibers. Nodes are colored according to the symmetry fiber partition. White color is used to show the nodes in trivial (consisting of only one node) fibers. The input trees of nodes 3 and 11 are isomorphic, however they are separated into two different fibers in order to avoid a fiber partition in which nodes without a common input are grouped together. This is a biological requirement.

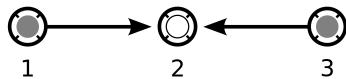


Fig. 13.15

Special cases of fibers. An example of the network in which fibrations considered in (Morone et al., 2020) are not minimal. Nodes 1 and 3 have no inputs and therefore belong to the same fiber of the minimal fibration by definition. However Morone et al. (2020) put them in separate fibers to fiber partitions in which nodes with no common input are put in the same fiber, since they are unlikely to synchronize in a real network.

Consider now the graph in Fig. 13.16. It is easy to see that the coloring produced by the algorithm assigns nodes 1, 3, 4, and 5 the same color. Indeed, all four nodes are assigned the same initial color, and corresponding ISCVs are the same. However, nodes 1 and 4 have no inputs in common with 3 and 5. Therefore, this coloring is not a realistic fiber partition from a biological viewpoint. The situation described here can be repaired using a further modification of the initial coloring. Let y_1, y_2, \dots, y_k be the set of sets of nodes belonging to the strongly connected components of the graph that receive no input, i.e., $y_i = j_1, j_2, \dots, j_{k_i}$, where j_m are nodes that belong to the strongly connected component

that receives no input with $id = i$. Then, the problem described here can be negated by assigning y_1, y_2, \dots, y_k different initial colors. This case appeared only once in the 373 networks studied in (Leifer et al., 2020). Therefore, we take it as a special case.

All these special biological cases are considered in Algorithm 5 showing the pseudocode employing these heuristics. An implementation of this algorithm in the form of an R package is available at <https://github.com/makselab/fibrationSymmetries>. See Appendix A for a comprehensive list of all codes for fibration analysis. We suggest the reader use this implementation in any study of a real-world biological network. However, a mathematician should adhere to the original minimal balanced coloring definition and ignore these special cases, i.e., the colorings of Fig. 13.15 and Fig. 13.16 should be considered the minimal.

As was mentioned at the beginning of this section, the Kamei–Cock algorithm is slower than the Paige–Tarjan algorithm. A faster implementation of the Paige–Tarjan algorithm to find symmetry fibers, called Fast Fiber Partitioning (FFP), is presented in (Monteiro et al., 2022).

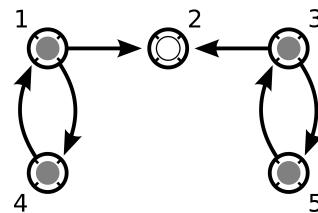


Fig. 13.16

Special cases of fibers. Example of a graph in which the algorithm does not produce the symmetry fiber partition. Nodes 1, 3, 4, and 5 have the same initial color and the same ISCVs. However, nodes 1 and 4 have no common inputs with nodes 3 and 5 and biology requires the two clusters to be of different colors.

PART II

APPLICATIONS: FROM GENES TO THE BRAIN

Part I developed the theoretical basis for the application of symmetries to biological networks. We saw that symmetries of the network come in different flavors. The more stringent symmetries are automorphisms with their related orbital partitions of synchronized nodes. Fibration symmetries and their associated clusters of synchronized fibers impose weaker conditions on the structure of the network since they preserve only input trees. Synchronized fibers are determined by the balanced colorings of the network. Part I provided theoretical evidence that fibration symmetry is necessary and sufficient to ensure cluster synchronization in the network. A primary objective of systems science is to decompose the network into independently functioning parts and scrutinize the emergence of the network dynamics from the interaction of these parts. Fibration symmetries provide a framework for this decomposition.

Part II focuses on biological applications of these theoretical methods. We show how symmetry fibrations describe symmetries and synchronization in biological networks including genetic networks, metabolic networks, signaling networks, and connectomes across species. We first consider two popular model organisms, *E. coli* and *C. elegans*, for which the most complete reconstructions of genetic, protein, and metabolic networks and the connectome, respectively, are known. Later we generalize our results to more complex organisms, including humans. We show that symmetries can directly predict gene synchronization activity of coexpression patterns observed experimentally, as well as neural synchronization in the brain. Furthermore, symmetry breaking of the synchronized fibers uncovers further building blocks with functionalities resembling memory devices of computation. The concept of a fibration can thus provide a full characterization of all genes in the genetic network, based on their symmetries and symmetry breaking. This provides a general organizing principle to understand the structure-function relation in biological networks.



Fibration Analysis of Biological Networks

Part I shows that cluster synchronization emerges from a symmetry fibration. This chapter shows that genetic networks exhibit symmetry fibrations, which group the genes in fibers with synchronized behavior, as shown in subsequent chapters. This creates a hierarchy of fibration building blocks of biological networks.

14.1 Geometric biology

Inspired by the symmetry theory of Part I, we now ask whether a geometrization process based on symmetry fibrations can be invoked to tame biological complexity, and if so, precisely how. We start by reviewing applications of symmetries to understand the structure of gene regulatory networks. Gene expression and its regulation have been studied since the time of the bacterium operon model by Jacob and Monod (1961); Jacob (1977). Gene products that cooperate to accomplish a certain task, for example, lactose utilization in the case of the Lac operon, tend to be coexpressed, i.e., differentially expressed in a synchronized manner. This genetic synchronization largely determines the function of the biological circuit. This leads to a hierarchy of fibration building blocks describing the geometry of the system.

14.2 Transcriptional regulatory networks

Systems biology relies on the construction of different types of biological networks including transcriptional, metabolic, signaling, protein-protein interactions and neural networks. The basic network that we analyze here is the transcriptional regulatory network (TRN) and the metabolic network of *E. coli*. This model bacterium *E. coli* possesses a genome with 4,690 genes, as compiled by RegulonDB's aggregate of results to the date of publication of this book (Tierrafría et al., 2022). Of these, 1,843 are known to have regulatory functions in the transcriptional network.

The transcriptional regulatory network is responsible for performing the decision-making process of how to change its gene expression levels according to the prevailing circumstances. For instance, organisms need to regulate the expression of their genes according to their (possibly changing) environment. To do so they must process information from transient environmental signals as inputs, and, along with the current state of expression levels,

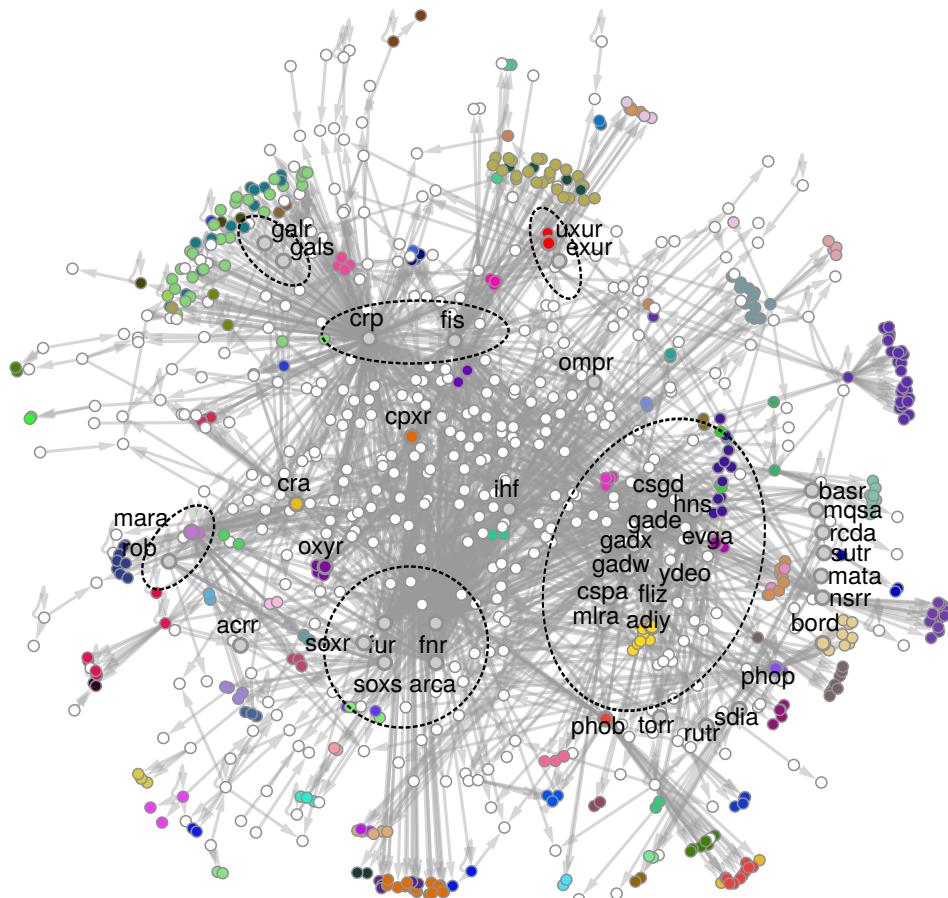
compute an appropriate response as an output, in the form of a new state of expression levels: namely, to decide which genes to turn off and which genes to turn on. In this manner they tune the expression level continuously as circumstances require.

Genes are regulated by transcription factors (TFs) that bind to their promoter region, which either activate the transcription of the target gene, ‘turning on’ the gene, or repress the transcription from happening, thus preventing expression ‘turning off’ the gene. The TFs are proteins coded by genes, so this amounts to a pairwise relation between genes, albeit indirectly: a gene codes for a TF, which then regulates other genes’ expression. The entirety of transcriptional regulations constitutes the transcriptional regulatory network (TRN) of the cell, which is in charge of deciding which genes to express. This endows the cell with the emergent property of being able to compute the next state of expression levels for the cell. The TFs are, in turn, usually activated or inhibited by the presence of specific small molecules, which function as input signals from the environment or the cell’s internal state. These small molecules, or metabolites, can either come from the environment or be the product of a metabolic pathway. Eukaryotes, as opposed to bacteria, have other forms of more complex regulations as well.

To see how signals propagate from gene to gene (in time or as a causal chain determining steady states), we need to consider the entire gene regulatory network. For simplicity, and due to limited knowledge about some regulation steps, we simplify the network and consider directed regulation edges between transcription factors and their target genes (all of them being described by protein levels). Neglecting many specific regulation steps (e.g., in transcription termination, ribosome binding, and mRNA or protein degradation), we describe the expression of a target gene by a simple *input function*, with the TF expression levels as function arguments.

Indeed, every transcriptional interaction between two genes (edge in the TRN) requires multiple parameters for a precise mathematical description of the genes’ expression dynamics (Klipp et al., 2016). These effective parameters capture the gene regulatory process from transcription, translation, protein folding, to binding and unbinding of the transcription factor to DNA, plus ribosome and polymerase binding, mRNA and protein lifetimes, to mRNA and protein degradation. This set of events is lumped into one single edge representing the whole process in the TRN, yet, this edge encapsulates a large number of parameters which are, in principle, unknown to the modeler trying to understand the regulatory process.

Since most of these parameters are unknown, structural graph analyses of these lumped networks have been performed by ignoring these heterogeneities and assuming that all edges are the same, i.e., each edge represents the same set of parameters, modulo some important differences such as repression and activation. (Chapter 21 examines in detail this assumption and offers alternatives.) Such an approach has led to considerable insights, in particular, the discovery of network motifs (Shen-Orr et al., 2002; Milo et al., 2002). Motifs are building blocks of the network that are identified by statistical over-representation of small circuits in the network, as compared to randomized networks. Their existence suggests that, as expected, the network structure has most likely been shaped by evolutionary pressures rather than being randomly connected. Even though the individual dynamics of some motifs is more or less understood (Shen-Orr et al., 2002; Milo et al., 2002), given that they are in

**Fig. 14.1**

The complexity of the *E. coli* TRN in its full glory. We show the weakly connected component of the operon-TRN for *E. coli* with 879 nodes. Small disconnected pieces of the network are not shown since they do not play a role in the network's dynamics. Genes in the same fibers are colored the same. Strongly connected components are marked by ellipses with their gene names. In this chapter, we unravel the origin of this complex structure one piece at a time. Figure reproduced from (Álvarez-García et al., 2025a).

essence, local structures, they do little to unravel the global topology of the network or the global dynamics. Also, statistical significance alone is not enough to explain function.

We construct a directed network for this process, representing all the transcriptional regulatory signals. A directed edge in the network represents a transcriptional signal from one gene to another, where the source of the edge represents the gene coding for the transcription factor that binds to the gene represented by the target node of the edge. The type of edge corresponds to the type of regulation the transcription factor performs, either

activation or repression. (Further distinctions, such as connection strengths or weights, can be made, but we ignore these refinements at this stage, see Chapter 21.) The full complexity of the *E. coli* TRN is shown in Fig. 14.1 highlighting the structures—strongly connected components, fibers and building blocks—that we unravel in this chapter.

The process of sending regulatory messages across genes defines the information flow in the transcriptional network, which we can think of as an ‘information package’ or ‘message passing’ from the source gene to the target gene. Thus, we interpret a link in a transcriptional network as representing regulatory ‘messages’ that are dynamically sent from a source gene to a target gene via the transcription factor. The transcription factor acts as a ‘messenger’ to repress or activate the transcription rate of the target gene. This information flow is not restricted to two interacting genes, but is transferred to different regions within the network that are accessible through the connecting pathways. The information arriving at a gene contains the entire history transmitted through all pathways that reach this gene, which is formalized by the input tree of the gene.

On a large scale, it has been proposed that the *E. coli* TRN has a feed-forward structure (Ma et al., 2004; Martínez-Antonio et al., 2008; Dobrin et al., 2004; Shen-Orr et al., 2002), where signals flow unidirectionally from a core of sensors and master regulators through a series of parallel layers down to an outer periphery in a feed-forward manner (Dobrin et al., 2004; Shen-Orr et al., 2002; Milo et al., 2002). However, there also exist feedback loops that complicate this picture. Thus, many questions remain unanswered:

1. What is the relevant structure of this system that determines its function?
2. What are the network’s building blocks?
3. What is the computational core structure in charge of decision-making in the cell?
4. How does this structure control the rest of the network?
5. Is there a minimal structure that explains the function of the TRN in a simplified manner?

Fibration symmetries offer possible answers to these questions.

14.3 Fibration analysis of the TRN

We search for symmetries in the *E. coli* transcriptional network using the compilation TF interactions at RegulonDB (Tierrafría et al., 2022). Nodes are genes; a directed link represents an activator, repressor (or sometimes a dual) transcriptional regulation mediated by a transcription factor.

We start our discussion by analyzing the symmetries in a sample sub-circuit extracted from the TRN of *E. coli*. This circuit is regulated by gene *cpxR* which regulates its own expression (via an autoregulation positive loop) and regulates other genes as shown in Fig. 14.2a. (The circuit regulates more genes, represented by the dotted lines, which are not displayed for simplicity and do not change the results.) Gene *cpxR* is not regulated by any other transcription factor, so it does not receive ‘information’ from the main network (we say that this gene forms its own ‘strongly connected component’, see below). Therefore, it is an ideal simple circuit to exemplify the concept of graph fibration in the TRN.

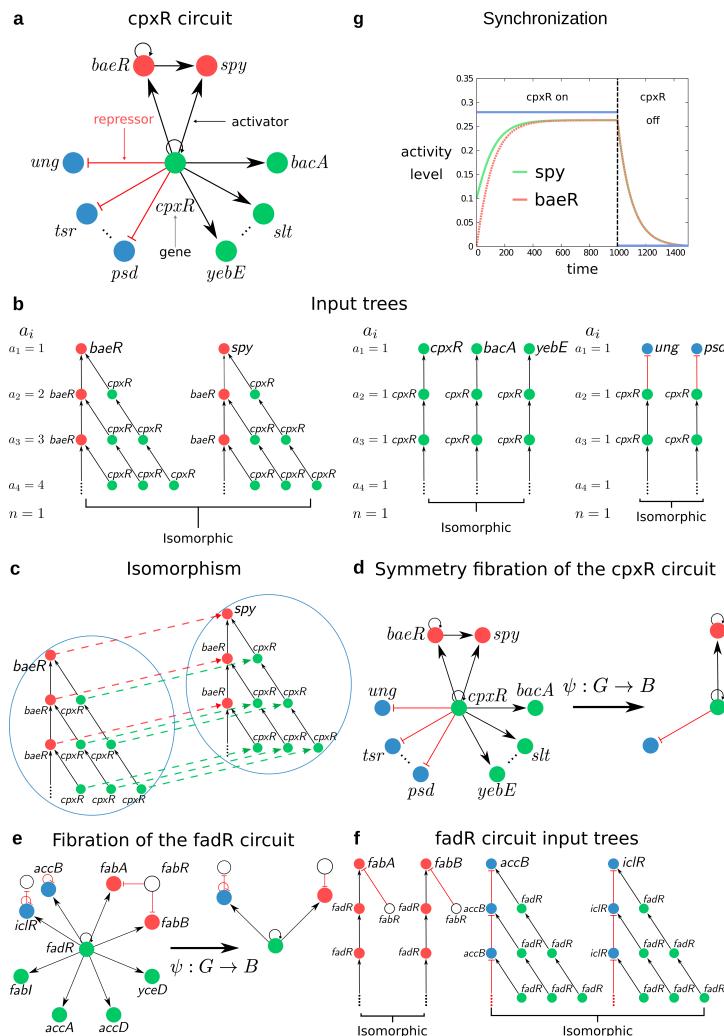


Fig. 14.2

Example of input tree, fibration, fiber, and base in a simple *E. coli* circuit. (a) The circuit controlled by the *cpxR* gene regulates a series of fibers, as shown by the different colored genes. (b) The input tree of representative genes involved in the *cpxR* circuit shows the isomorphisms that define the fibers. (c) Isomorphism between the input trees of *baeR* and *spy*. (d) Fibration ψ transforms the *cpxR* circuit G into its base B by collapsing the genes in the fibers into one. (e) Fibration of the *fadR* circuit and (f) its isomorphic input trees. (g) Symmetric genes in the fiber receive the same information through the pathways in the network and, therefore, synchronize their activity to produce activity levels leading to a common cellular function. Figure reproduced from (Morone et al., 2020). Copyright © 2020, National Academy of Sciences U.S.A.

The input tree of gene *spy* (Fig. 14.2b) starts with *spy* at the root (first layer). Since this

gene is upregulated by *baeR* and *cpxR*, the second layer of the input tree contains these two pathways of length one starting at both genes. Gene *baeR* is further upregulated by *cpxR* and by itself through the autoregulation loop, and *cpxR* is also autoregulated. Thus, the input tree continues to the third layer, taking into account these three possible pathways of length 2: one starting at *baeR*, and two starting at *cpxR*. The procedure now continues, and since there are loops in the circuit—the autoregulation loops at *baeR* and *cpxR*—the input tree has an infinite number of layers.

As discussed in Section 5.6, the theorem of Norris (1995) allows us to prove the equivalence of two input trees with an infinite number of levels (or layers). It suffices to find an isomorphism up to $N - 1$ levels, where N is the number of nodes in the circuit. Since the ribose system contains three genes, only input trees up to the second level are needed to prove the existence of an isomorphism. In fact, the input trees of *rbsR* and *rbsDACKB* are isomorphic, but not with *crp*.

The input trees in the *cpxR* circuit are displayed in Fig. 14.2b. The input trees of *baeR* and *spy* (Fig. 14.2c) are isomorphic and define the *baeR-spy* fiber with an isomorphism:

$$\begin{aligned}\tau = & (baeR, baeR, cpxR, baeR, cpxR, cpxR, \dots) \\ \rightarrow & (spy, baeR, cpxR, baeR, cpxR, cpxR, \dots).\end{aligned}\quad (14.1)$$

Clearly, the input tree of *cpxR* is not isomorphic to either *baeR* or *spy*, and therefore *cpxR* is not symmetric with these genes, but it is isomorphic to *bacA*, *slt* and *yebE* forming another fiber. Likewise, *ung*, *tsr* and *psd* are all isomorphic forming their corresponding fiber (Fig. 14.2b). Figure 14.2d shows the fibration $\psi : G \rightarrow B$ that collapses the genes in the fibers to the base *B*. Figure 14.2e shows another example (out of many) of a single connected component, *fadR*, and its corresponding isomorphic input trees (Fig. 14.2f), fibers and base.

In Fig. 14.2g we use the mathematical model of gene regulatory dynamics of Boolean kinetics from (Shen-Orr et al., 2002) to show synchronization inside the fiber *baeR-spy* when the fiber is activated by its regulator *cpxR*. Notice that *cpxR* does not synchronize with the fiber. Sigmoidal interactions lead to qualitatively similar results.

Going beyond these particular examples, Morone et al. (2020) performed a full fibration analysis of this biological network. The first step is identifying the strongly connected components in the system, because the classification of fibers is based on the cycles that the fibers form.

Recall the concept of a strongly connected component, Definition 3.8: a directed graph is strongly connected if there is a (directed) path between all pairs of vertices; a strongly connected component (SCC) is a maximal induced subgraph that is strongly connected. As stated earlier, we may omit the adjective ‘strongly’ when this does not cause confusion.

Beyond several single-gene strongly connected components like those shown in Fig. 14.2, the *E. coli* network has three main SCCs, which regulate a set of fibers:

- **Sugar SCC:** A two-gene feedback loop strongly connected component composed of master regulators *crp-fis* involved in a myriad of functions related to carbon utilization and sugar metabolism (Fig. 14.3).
- **pH regulation SCC.** The largest SCC at the core of the network which is composed of

2a cpr-fis connected component

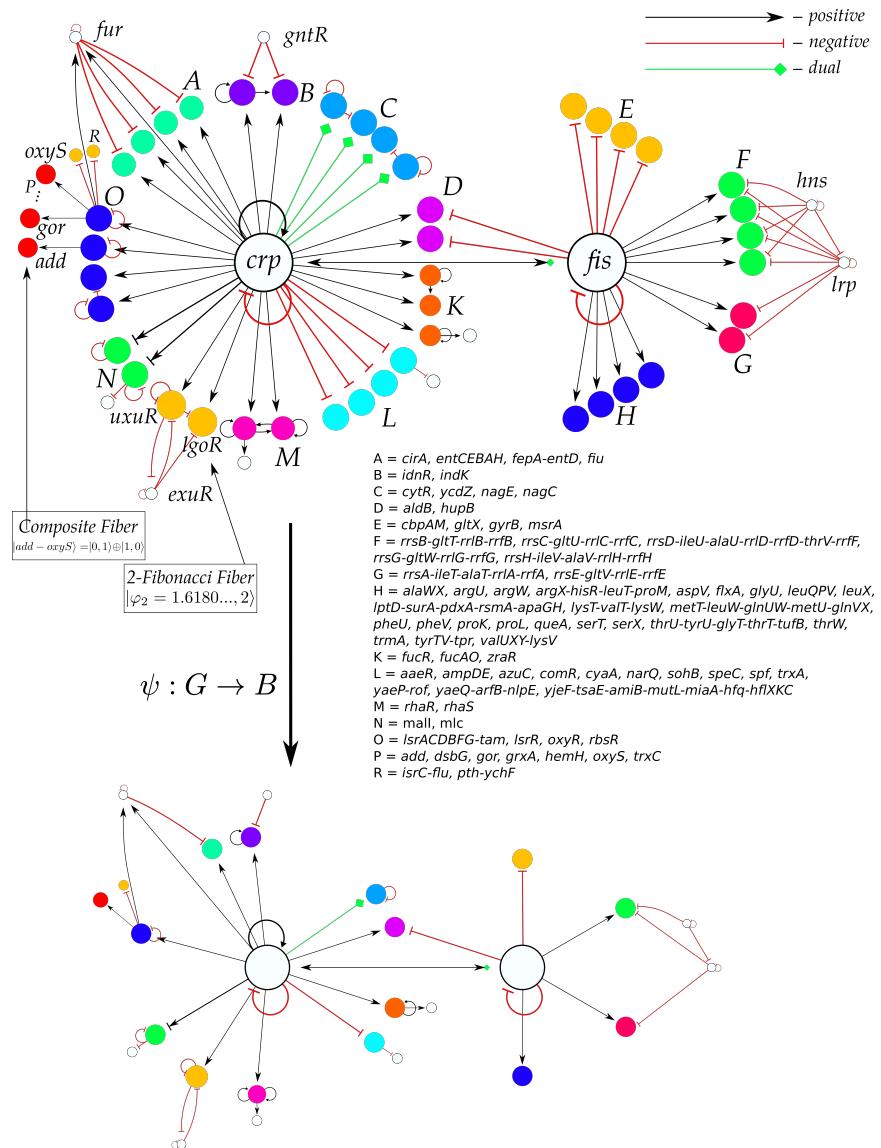


Fig. 14.3

Fibers of the strongly connected components of sugar metabolism in *E. coli*

TRN. Two-gene connected component of *cpr-fis*. This component controls a rich set of fibers as shown. We also show the symmetry fibration collapsing the graph to the base. We highlight the fiber *uxuR-lgoR* which sends information to its regulator *exuR* and forms a 2-Fibonacci fiber $|\varphi_2 = 1.6180\ldots, l = 2\rangle$, as well as the double-layer composite $|add - oxyS\rangle = |0, 1\rangle \oplus |1, 1\rangle$. We denote the fibration symmetry transformation to its base by ψ . Figure reproduced from (Morone et al., 2020). Copyright © 2020, National Academy of Sciences U.S.A.

genes involved in the pH-system that regulate hydrogen concentration inside the cell, such as *GadX*, *GadW*, *Hns*, *GadE*, *GadF*, *YdeO*, *EvgA*, *FliZ* (Fig. 14.4).

- **Stress regulation SCC.** A five-gene strongly connected component involved in the stress response system (Fig. 14.5).

Applying the fiber finding algorithm based on the minimal balanced coloring of the network from Section 13.7 we find a rich set of fibers that are regulated by each of the three components. These fibers are then collapsed into the base by the symmetry fibration $\psi : G \rightarrow B$, as shown in the figures.

Applying the symmetry fibration to the *E. coli* TRN leaves just 555 nodes in the base, 30% the size of the original network, see Table 14.1. Figure 14.6 show a sample of the fibers in different functional circuits such as the non-glucose carbon production circuit regulated by *crp*, and amino-acid production fibers. A great variety of fibers is observed. They are classified below into an hierarchy of fibration building blocks.

As shown in Table 14.1, the full *E. coli* genome contains 4,690 genes, out which 1,843 genes express TFs that are not isolated, i.e. they are TFs with at least a regulation to or from another TF. Bacterial genomes contain operons, which are sets of continuous genes that are transcribed together into mRNA by the same RNA polymerase sharing the same promoter (the part of DNA that enables the gene to be transcribed). As a result, operons are expressed together, so they are trivially synchronized. But not always: some operons are split into Transcriptional Units (TU) which are operated by different promoters. These promoters and end points are intercalated between the genes of the operon. Thus sometimes the operon is partially expressed, and sometimes it is fully expressed. So synchronization in operons is a bit complicated, but in general, pure operons (which do not contain TUs) are considered to be fibers that synchronize trivially via a common polymerase.

Interestingly, these operons are fibers, albeit of a trivial shape since they are trivially synchronized (Section 15.8). To simplify the analysis we just collapse them by a trivial fibration and then analyze the remaining network, which in this case contains 879 TFs (Table 14.1). In this network, 63% of the genes are involved in fibers, indicating a large coverage of fibers highlighting their importance for the structure of the network. The remaining genes are mainly dangling-ends, genes that send information to other parts of the cellular network like the metabolism. When studying the TF, these genes can be removed, and the remaining network contains just 42 genes from the original 1,843. This reduced network still preserves the same dynamics as the original network, on account of the invariance of the dynamics under fibration, with all redundant information pathways collapsed into a single one. This remarkable reduction highlights the power of fibrations to tame biological complexity. It is what we call the ‘minimal genome’, studied with more detail in Chapter 19.

Most of the fibers regulated by these components do not belong to the connected components, although they are regulated by them. This is because they receive information from the connected component, but do not send information back to it. This implies that we can classify these fibers with simple integer dimensions, as we see next. More interestingly, fibers send information to the connected components, creating cycles that produce fractal dimensions (see Section 15.7.1) with more interesting memory dynamics.

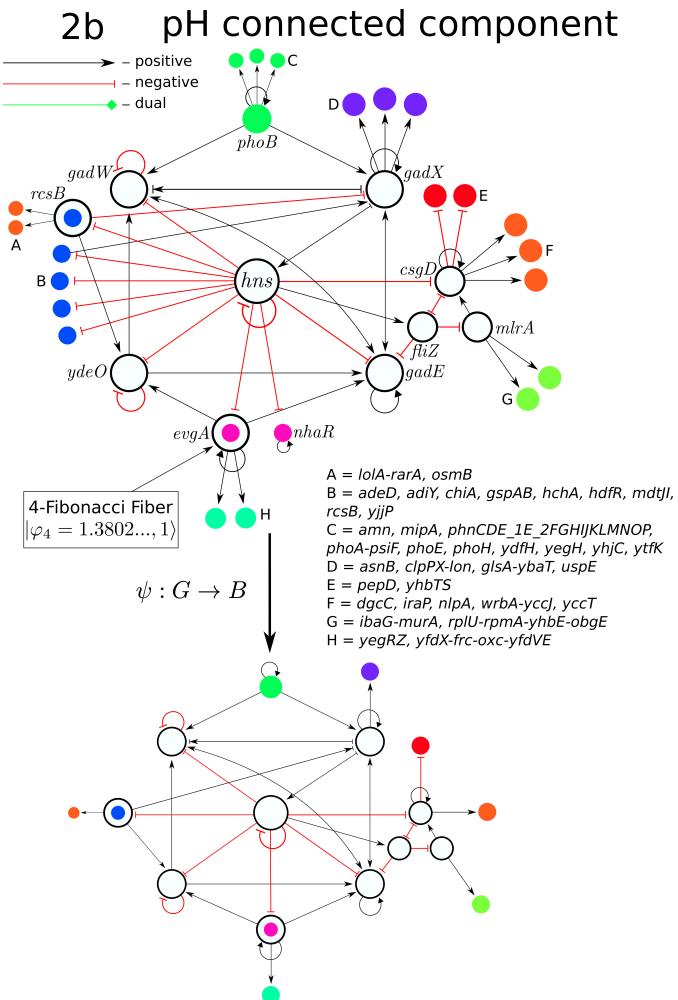


Fig. 14.4

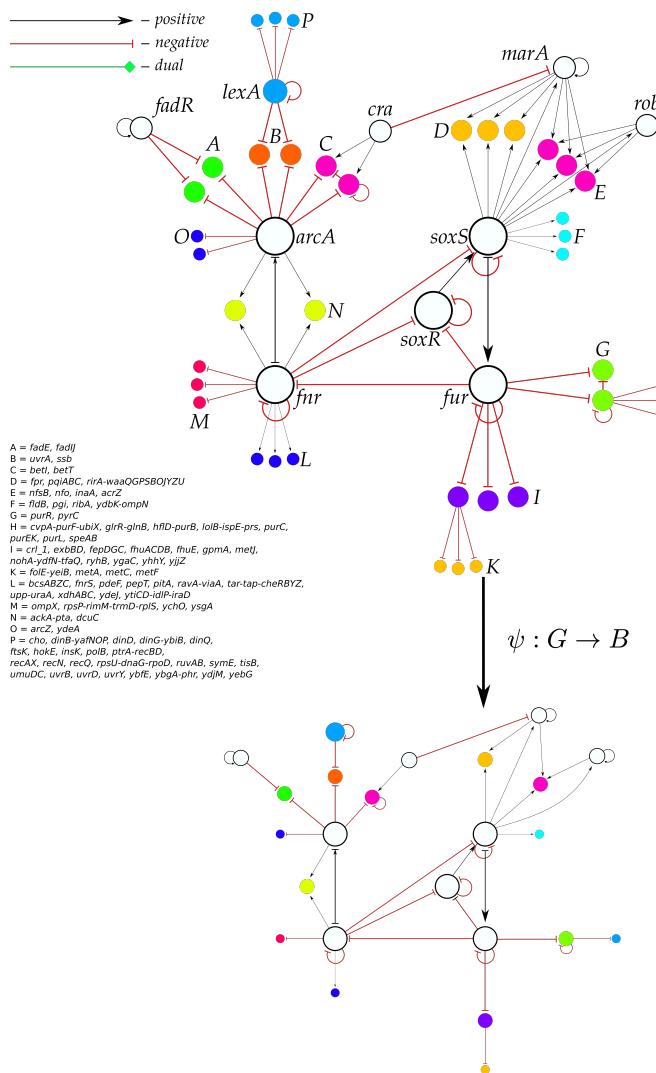
Fibers of the strongly connected components of the pH regulation in *E. coli* TRN.

The core of the *E. coli* network is the strongly connected component formed by genes involved in the pH system as shown. This component supports two Fibonacci fibers: 3-FF and 4-FF and fibers as shown. Hollow colored circles indicate genes that are in fibers and also belong to the pH component. Figure reproduced from (Morone et al., 2020). Copyright © 2020, National Academy of Sciences U.S.A.

14.4 Searching for biologically meaningful building blocks

In a gene network, how can we find (possibly overlapping) groups of genes that form ‘meaningful circuits’, i.e. circuits with a specific biological function? The function might

soxR-soxS-fnr-fur-arcA connected component

**Fig. 14.5**

Fibers of the strongly connected components of the stress response in *E. coli* TRN. A five-gene connected component of *soxR*, *soxS*, *fnr*, *fur*, and *arcA* with its regulated fibers. Figure reproduced from (Morone et al., 2020). Copyright © 2020, National Academy of Sciences U.S.A.

be performing a specific signal processing task such as spike generation, filtering, forming logical gates, etc.; or it might be regulating a specific biological system, e.g. in response to external stimuli or via feedback regulation. Circuits are likely to be meaningful if (i) they create a meaningful dynamics or perform a meaningful regulation of signal processing tasks, for example coexpression of target genes; (ii) circuits of this shape occur many times

Reduction step	Gene count	%
Full Genome RegulonDB	4,690	—
TRN (non isolated TFs)	1,843	100%
Operon-collapsed TRN	879	48%
Base-TRN (collapsing fibers)	555	30%
Minimal TRN (after trivial pruning)	42	2%

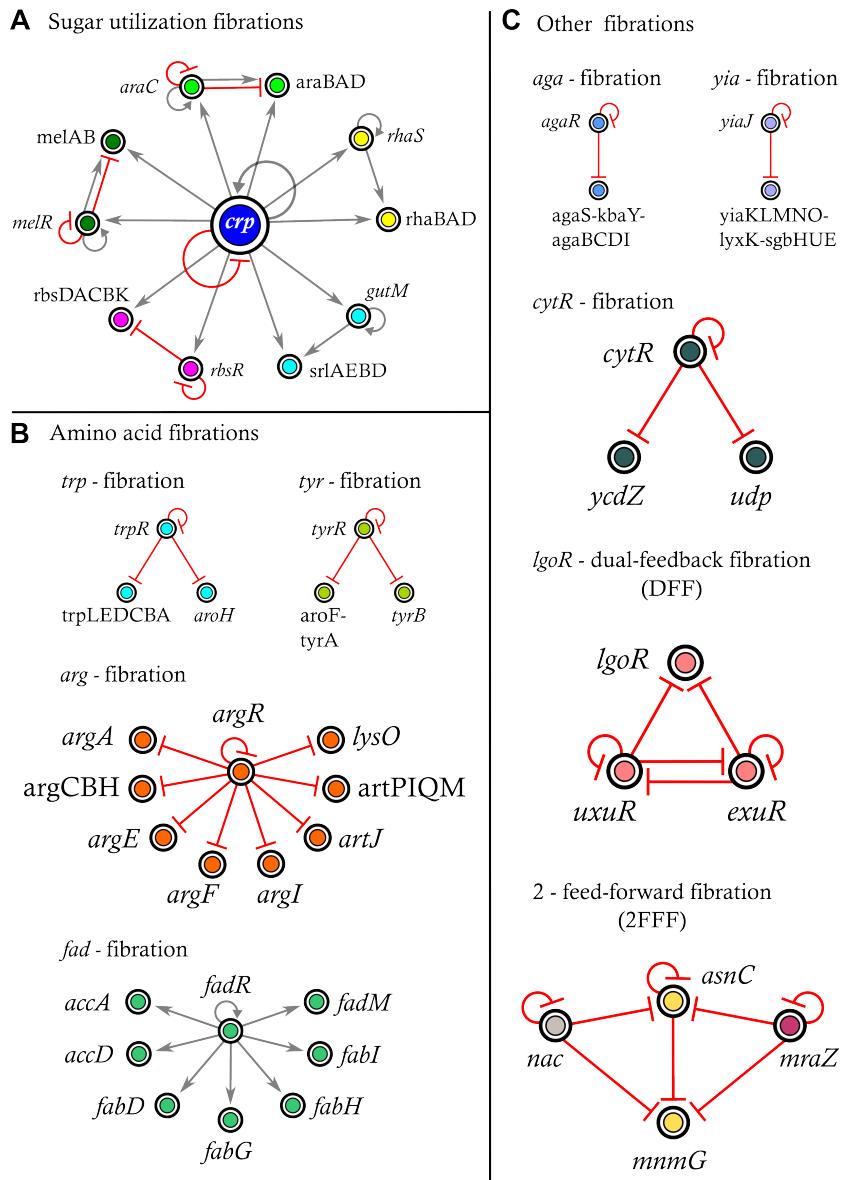
Table 14.1: **Reduction count and fiber coverage.** The full *E. coli* genome contains 4,690 genes, of which 1,843 genes express TFs that are not isolated, i.e. they are TFs with at least a regulation to or from another TF. The first reduction of operons is by a trivial fibration—which collapses the operons = trivial fibers, reducing the network to 879 TFs. The fibration further reduces the network to 30% and after removing the trivial dangling-ends, and the final base is reduced to 2% of the original size conforming the minimal TRN. Table reproduced from (Álvarez-García et al., 2025a).

in a network, much more often than expected by chance (network motifs); (iii) they are biochemically implemented through structures like graph fibers.

Searching for these circuits as building blocks underlying the structural organization of biological networks is one of the cornerstones of systems biology (Hartwell et al., 1999a; Alon, 2019; Klipp et al., 2016). In the framework of network science, an enormous variety of network metrics have been applied to biological networks, neural circuits, and structural and functional brain networks in search of their fundamental building blocks. At the sub-circuit level the motifs introduced by Alon and collaborators (Shen-Orr et al., 2002; Milo et al., 2002; Alon, 2019), are the most widely used building blocks in biological networks. These are small subgraphs of the network, the most prominent being the feed-forward loop, which are over-represented as compared to a null model (defined by randomly rewiring the original network while preserving the degree sequence). We define motifs more formally in Chapter 17.

This approach is blind to the fact that gene outputs depend on the entire previous signal processing, which may involve loops. Mathematically, the dynamics of the motif in isolation need not correspond to its dynamics when embedded in the surrounding network, because inputs to the motif affect its behavior. Over-representation can be a useful indicator of potential functionality, but of itself it carries no implications for dynamics. Thus it is questionable whether network motifs would have functionality when embedded in the network. Furthermore, genes may exhibit similar expression patterns, even if they do not share the same inputs, as shown in Section 20.4.2.

Beyond motifs, there is a vast literature on frameworks use for finding meaningful network structures based on ideas like small-world (Watts and Strogatz, 1998), scale-free (Barabási and Albert, 1999), modularity (Girvan and Newman, 2002), community detection algorithms (Blondel et al., 2008), principal components, clustering, and centrality analysis (Makse and Zava, 2025) based on hubs (Barabási and Albert, 1999), rich-clubs, Pagerank, eigenvector centralities, k -core (Kitsak et al., 2010), or collective influence (Morone and Makse, 2015; Del Ferraro et al., 2018), to name a few. These network properties, while

**Fig. 14.6**

Assorted sample of fibers in the TRN. (a) The non-glucose sugar utilization system contain many fibers with *crp* acting as the master regulator switch without synchronizing with any fiber. (b) Sample fibers responsible for amino-acid production in the cell. (c) More elaborate fibers.

providing a summary description of the structure of the networks, are too generic to illuminate the functionality underlying biological networks, since they cannot be traced to fundamental principles.

Even the statistical evidence has been criticized. Fodor et al. (2020) argue that common methods for searching for motifs involve testing a large number of hypotheses in parallel, creating a high risk of false positives. They consider two standard assumptions: a normal distribution, and independence between candidates. They find cases where both of these assumptions are violated, and suggest that this casts doubt on some mainstream motif identification techniques. Stivala and Lomi (2021) suggest the use of exponential random graph models to overcome such statistical shortcomings.

Furthermore, there is no experimental evidence for a link between network motifs and dynamics (Ingram et al., 2006), and controversies remain on the functional role of motifs in biological networks (Ingram et al., 2006; Payne and Wagner, 2015; Macía et al., 2009; Ahnert and Fink, 2016). These criticisms suggest that network motif structure does not correlate with function.

In contrast to network motifs, which are found by statistical over-representation in the network, gene fibers reveal meaningful genetic circuits with functional relevance using theory. These fibers are relevant even if they appear only once in a network. Indeed, while motifs can be found on statistical grounds through their number of occurrences, functional circuits that do not occur in large numbers may easily go unnoticed. This remark implies, in particular, to larger circuits, for which statistical over-representation as a motif (by counting all subgraphs of that size) or as a module (Hartwell et al., 1999a) (by counting a ‘density’ of edges within that module) is hard to prove.

Fibrations and their associated fibers provide a definition of biological building blocks using first physical principles, rather than by statistical significance. We use the principle of symmetry, since it directly predicts synchronization, no matter how often these structures are found in the network. The resulting circuits can perform signal processing tasks, namely generating structurally encoded, yet tunable patterns of coexpression.

In the coming sections, we apply fibration theory to gene regulatory networks across species to find and classify the fibration building blocks of biological networks. We then validate the predicted coexpression of genes in the fibers of the gene regulatory networks of bacteria. We also show that the fibration building blocks are enriched by functional annotations and thus are biologically significant in the cell. We compare the results with motifs in more details in Chapter 17.

14.5 Definition of fibration building blocks

The dynamical state of a gene is encoded in the topology of the input tree. In turn, this topology is encoded by a sequence, a_i , defined as the number of genes in each i -th layer of the input tree, see for instance Fig. 14.2b. The sequence a_i represents the number of paths of length $i - 1$ that reach the gene at the root. This sequence is characterized by the branching ratio r of the input tree defined as:

Definition 14.1 Branching ratio of the input tree. Given a_i as the number of source genes with paths of length $t - 1$ to the target gene in the input tree, the *branching ratio* r is

defined as:

$$r = \lim_{i \rightarrow \infty} \frac{a_{i+1}}{a_i}, \quad (14.2)$$

provided this limit exists, see Remark below.

Another term for the branching ratio, which is common in the literature and which we use on occasion throughout the book, is ‘fractal dimension’.

Remark The limit (14.2) need not exist, in which case a more general definition is required. An example is Fig. 6.7a, the input tree of the 6-node network of Fig. 4.2a. In this case, the sequence a_i is:

$$1, 1, 2, 2, 2, 4, 4, 4, 8, 8, 8, \dots$$

The ratios a_{i+1}/a_i are:

$$1, 1, 2, 1, 1, 2, 1, 1, 2, \dots$$

which fails to converge. Such networks are rare and unlikely to occur in biology. When the limit in 14.2 does not exist, we can replace it by

$$r = \lim_{i \rightarrow \infty} a_i^{1/i} \quad (14.3)$$

The Perron–Frobenius Theorem implies that this limit always exists, and is equal to the spectral radius (here the maximal eigenvalue, which is real) of the adjacency matrix of the ‘upstream subnetwork’: the induced subgraph for all nodes connected to i by a path. It takes the same value as (14.2) when that limit exists. See (Boldi and Stewart, 2025).

The branching ratio represents the ‘average long-term’ multiplicative growth rate of the number of paths across the network reaching the gene at the root. For instance, the input trees of genes *baeR-spy* (Fig. 14.2b) encode a sequence $a_i = i$ with branching ratio $r = 1$ representing the single autoregulation loop inside the fiber.

The branching ratio organizes the topologies of the fibers in a neat fashion. For instance, the TRN of *E. coli* is organized into 91 different fibers. The complete list of fibers in *E. coli* appears in the supplementary information section of (Morone et al., 2020). We find a rich variety of topologies of the input trees. Despite this diversity, the input trees present common topological features that classify all fibers into concise classes of fundamental ‘fibration building blocks’.

The intuition for defining building blocks is as follows. The central component of the building block is the fiber. Since the genes in the fiber are synchronous, the building block captures the idea of common functionality of the genes by associating function with coherence. This is the most natural notion of a building block. However, the fiber does not work in complete isolation since it receives signals from the network, in particular from the SCC to which it belongs. (It also sends signals to it.) Thus, it is common sense to add to the central fiber of the building block the external regulators of the fiber. In the simplest case, a fiber just receives signals from its regulator but does not send signals back to the regulators nor to any other part of the network. In this case, the definition of building block is unambiguous. We associate a building block with this fiber by considering the genes in the fiber plus the external incoming regulators of the fiber.

For instance, the fiber denoted by B in the carbon SCC in Fig. 14.3:

$$\text{Fiber} = \{idnR, indK\} \quad (14.4)$$

receives signals from

$$\text{Regulators} = \{crp, gntR\} \quad (14.5)$$

and do not send any signals outside the fiber. Then, the building block is *simple* and it is formed by:

$$\begin{aligned} \text{Simple Fibration Building Block Level 1} &= \text{Fiber} + \text{Regulators} \\ &= \{idnR, indK, crp, gntR\}. \end{aligned} \quad (14.6)$$

We call it a simple level 1 building block since, from here, we can build more complex ones. The next level of complexity appears when the fiber sends signals back to the regulators or to other genes in the SCC or to any other part of the network through more than one path. This creates cycles that are important to define the dynamics of the fiber. In principle, all these cycles should be included in the definition of the building block associated with the central fiber.

When the fiber sends signals back to the regulators through a *unique* path in the network from the fiber to the regulator, that is when there is only a feedback loop between fiber and regulators, then we consider the building block still *simple* but level 2. It is given by (14.6). For instance, the yellow fiber in Fig. 14.3:

$$\text{Fiber} = \{uxuR, lgoR\} \quad (14.7)$$

is still a simple building block because it has a single cycle from fiber passing to one of its regulators *exuR*. We call this particular building block Simple Fibonacci building block and will be studied later:

$$\begin{aligned} \text{Simple Fibration Building Block Level 2} &= \text{Fiber} + \text{Regulators} \\ &\quad (\text{with Unique Path Through Regulators}) \\ &= \{uxuR, lgoR, exuR, crp\}. \end{aligned} \quad (14.8)$$

The majority of fibers in the TRN of *E. coli* shown in Figs. 14.3, 14.4 and 14.5 are simple Level 1 and 2 and Eqs. (14.6) and (14.8) suffices to define them.

However, the fiber

$$\text{Fiber} = \{evgA, nhaR\} \quad (14.9)$$

denoted in red in the pH SCC of Fig. 14.4 clearly presents a more complex situation. This fiber (actually only the gene *evgA*) is embedded in the SCC, implying that there could be more than one cycle, starting from the fiber and passing through the regulators. In principle, the genes belonging to these paths should be included in the building block of the fiber since all the genes in these cycles are crucial to establish the dynamic synchronization state of the fiber. These cycles can be complex and long, and if the fiber is embedded in the SCC, they may even include all cycles in the SCC.

At this point, we introduce a level of subjectivity into the definition of a building block: we consider as part of the building block only the shortest path. We call these building

blocks *simple* level 3. We use this definition of building block elaborated in (Morone et al., 2020; Leifer et al., 2020).

For instance, the *evgA* fiber has a regulator:

$$\text{Regulator} = \{hns\} \quad (14.10)$$

that belongs to the SCC of *evgA* and *evgA* is connected to the SCC. Thus, there are pathways through the SCC that connect the regulator and the fiber. Two of them are:

$$\text{Cycle 1} = \text{evgA} \rightarrow \text{gadE} \rightarrow \text{gadX} \rightarrow \text{hns} \rightarrow \text{evgA} \quad (14.11)$$

$$\text{Cycle 2} = \text{evgA} \rightarrow \text{ydeO} \rightarrow \text{gadE} \rightarrow \text{gadX} \rightarrow \text{hns} \rightarrow \text{evgA}. \quad (14.12)$$

Since Cycle 1 is the shortest, only this path is included in the building block of *evgA*. Not all genes in the SCC participate in these paths. Where there are many shortest paths with the same length, we either pick one at random or consider all of them. We call this a:

$$\begin{aligned} \text{Simple Fibration Building Block Level 3} &= \text{Fiber + Regulators} \\ &\quad + \text{Shortest Path Through Regulators} \\ &= \{\text{evgA}, \text{ydeO}, \text{gadE}, \text{gadX}, \text{hns}\} \end{aligned} \quad (14.13)$$

All simple building blocks are regulated by isolated genes. The next level of complexity appears when the fiber is regulated by another fiber. This increases the complexity of the building block massively since there are many paths between the fibers, creating cycles of information that increase the fractal dimension of the input trees. We call these building blocks *complex*. They are abundant in the metabolic networks of Chapter 16 and complex organisms.

$$\begin{aligned} \text{Complex Fibration Building Block} &= \text{Fiber + Fiber Regulators} \\ &\quad + \text{Shortest Path Through Fiber Regulators} \end{aligned} \quad (14.14)$$

We formalize these intuitions in the following definition (Morone et al., 2020; Leifer, 2022).

Definition 14.2 Fibration building block. A *fibration building block* is a subgraph induced by the set of nodes consisting of:

1. All the nodes in the fiber.
2. All regulators that send inputs to the fiber.
3. All nodes belonging to the shortest cycle, including a node in a fiber if any node in a fiber is a part of a cycle.

For example, consider the network in Fig. 14.7 with a fiber in green. The building block corresponding to the fiber consists of 5 nodes: (a) nodes 1, 2, 3 belong to the fiber, (b) node 4 regulating this fiber, and (c) node 5 belonging to the shortest cycle that include a node in the fiber. Despite the existence of the cycle 3 → 5 → 6 → 4 → 3, node 6 is not included in the building block because this cycle is not the shortest.

A final distinction should be made between the input trees of the fiber and the input trees

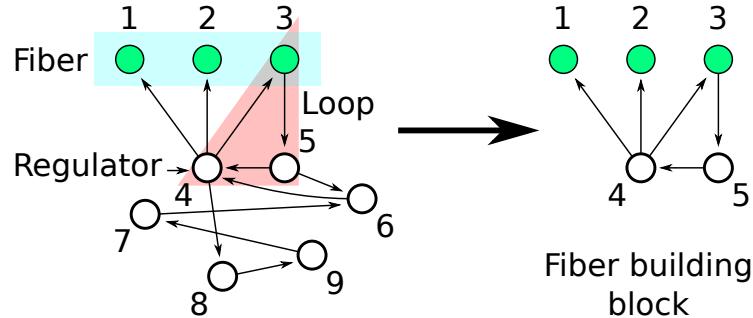


Fig. 14.7

Example of a fibration building block (Leifer, 2022). The building block corresponding to the fiber in green consists of $Fiber = \{1, 2, 3\}$, fiber regulator $\{4\}$, and a node in the shortest cycle $\{5\}$ going through the regulator 4. Longer cycles are neglected.

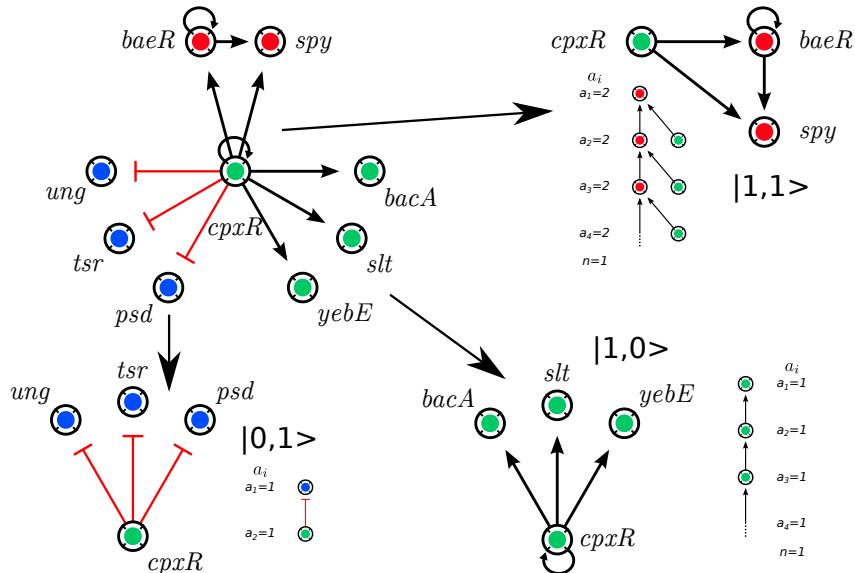
of the building block. We exemplify this case with the fiber $G = \{purR, pyrC\}$ in the stress SCC shown in Fig. 14.5. The fiber and building block and input trees of the building block are shown in Fig. 15.4. This is the feed-forward fiber (FFF) already discussed in Sections 5.1, 5.5, and 8.6.

This building block is embedded in the network, but the building block ignores the paths reaching the regulator *fur*. For instance, *fur* receives signals from *soxS* and itself in the SCC. Moreover, it also receives inputs from *oxyR*, although this is not shown in Fig. 14.5 since *oxyR* does not belong to the SCC. However, all these regulators of *fur* (and the paths that comprise its input tree) are not crucial to determine whether *purR* and *pyrC* are synchronized, since the regulators of *fur* are not connected to, nor receive any inputs from, the fiber genes. We do not consider the autoregulation loop at *fur* to be in the building block, either.

Thus, the input trees of the building block are those of the circuit composed of *fur-purR-pyrC* without considering any inputs of *fur* (including its own autoregulation). The input tree of *fur* as considered in the building block is empty, as shown in Fig. 15.4b, while the input tree of *fur* as embedded in the network is obviously more complex. The same consideration applies to the input trees of the genes in the fibers. The full input tree of *purR* and *pyrC* in the network should be augmented by the paths reaching *fur*. These larger input trees remain isomorphic, demonstrating that the paths beyond *fur* do not affect synchronization of the fiber. We conclude that the minimal circuit that guarantees synchronization is what we call the building block.

More subtle differences are exemplified in the building block decomposition in Fig. 14.8 of the component regulated by *cpxR* that we encountered before in Fig. 14.2. This circuit is simple because it does not receive any signal from other genes. Yet, there are some subtleties.

The central regulator *cpxR* plays the role of regulator for the red fiber *baeR-spy* and the

**Fig. 14.8**

Example of fiber building block decomposition and classification. The presented network consists of three fibers and contains three corresponding building blocks (Leifer, 2022). The fiber building block corresponding to the nodes in green consists of nodes in the green fiber and no other nodes because this fiber is not regulated externally. The input tree of this fiber is an infinite chain with $a_i = 1$. Therefore this fiber building block is classified as $|n = 1, l = 0\rangle$. Similarly, the input tree of the fiber in red consists of an infinite chain with the addition of the external regulator, rendering this block as $|n = 1, l = 1\rangle$. The fiber building block in red has a finite input tree and one external regulator, and is therefore classified as $|n = 0, l = 1\rangle$.

blue fiber containing *ung*, but it belongs to the green fiber containing *bacA*. While the fibers form a non-overlapping partition, the building blocks can overlap. The input tree of the building block of *baeR-spy* does not include the autoregulation loop at *cpxR*. This encodes a sequence $a_i = 2$ for all i , with branching ratio 1 representing the single autoregulation loop inside the fiber and one external regulator.

The input tree of the building block of genes *ung*, *tsr* and *psd* is finite because the autoregulation loop in the external regulator *cpxR* is not included in the building block. This fiber is also an orbit, as it arises from an automorphism permuting the genes. In a sense, this is a trivial fiber, and the inputs suffice to determine its synchronization either by fiber or by orbit. The other two fibers are nontrivial, since they are not group symmetric. On the other hand, *cpxR* appears as a part of the green fiber *bacA* and its loop is included in the input tree of the genes in the building block.

In conclusion, we make a distinction between three different circuits:

- First, the actual fiber made of synchronized genes, like *purR-pyrC* in Fig. 15.4a.
- Second, the building block associated with the fiber as in Fig. 15.4a, which also includes the regulator: *fur*, *purR-pyrC*.
- Third, the fiber as embedded in the full network as in Fig. 14.5 with the added complexity of the input trees, i.e., with *fur* receiving signals from the rest, yet, representing information that is useless for understanding the dynamics of the fiber.

These considerations simplify the complexity of a building block to its minimal description, permitting a systematic classification of them all. It lets us obtain a concise topological classification of all fibers in terms of the fiber numbers, as we show next. Otherwise, if we were to include in the building blocks the paths beyond the regulator or other longer paths, a simple classification would not be possible.

14.6 Classification of simple building blocks

We provide a hierarchical classification of building blocks following (Morone et al., 2020). Using symmetries, we implement a constructive procedure that reveals a hierarchy of biological building blocks, ubiquitous across species. These conclusions introduce a theoretically principled strategy to search for computational building blocks in biological networks, as well as a route for conceiving synthetic biological circuits. We apply the algorithms to find fibers explained in Section 13.7 to find all fibers and then classify them into a hierarchy of building blocks in the TRN of *E. coli*. We start with the *simple* building blocks that describe most of the *E. coli* TRN, and then move on to describe a myriad of *complex* building blocks that appear in the metabolism and more complex species.

While we find a rich variety of topologies for input trees, we observe that input trees present common topological features that classify all fibers into concise classes of fibration building blocks. The key to developing a building block classification based on fibrations is to understand the hierarchical organization of loops and cycles, inside the fibers and between fibers and regulators. At the simplest level this is captured by the lengths of the cycles inside the fiber. These cycles determine the topology of input trees, hence the dynamics of the fiber.

The topological features of the fibers are encoded by the sequence a_i of the number of nodes in each i -th layer of the input tree. The sequence a_i represents the number of paths of length $i - 1$ that reach the gene at the root. The most basic input tree topologies and corresponding sequences a_i can be classified by integer ‘fiber numbers’.

Definition 14.3 Fiber number classification of simple fibration building blocks.
Simple building blocks are characterized by two *fiber numbers*:

$$|n, l\rangle : \text{simple building block classification} \quad (14.15)$$

Building Block	Number in <i>E. coli</i>
$ n = 0, l = 1\rangle$	45
$ n = 0, l = 2\rangle$	13
$ n = 0, l = 3\rangle$	3
$ n = 1, l = 0\rangle$	13
$ n = 1, l = 1\rangle$	8
$ n = 1, l = 2\rangle$	3
$ n = 2, l = 0\rangle$	1
$ n = 2, l = 1\rangle$	1
$ \varphi_d = 1.3802\ldots, l = 1\rangle$	1
$ \varphi_d = 1.4655\ldots, l = 1\rangle$	1
$ \varphi_d = 1.6180\ldots, l = 2\rangle$	1
Simple Multilayer Fiber	1
Total number of building blocks	91

Table 14.2: **Building block statistics.** We show the count of every building block in *E. coli* TRN defined by the fiber numbers. Table reproduced from (Morone et al., 2020).

reflecting two features:

- The integer n is the branching ratio $r = n$ of the input tree defined in (14.2), and represents the exponential growth of the number of paths through the network reaching the gene at the root.
- The number of finite pathways starting at l external regulators of the fiber. By convention, $n = 0$ for finite input trees.

For instance, the input trees of genes *baeR-spy* in Fig. 14.8 have branching ratio $n = 1$ representing the single ($n = 1$) autoregulation loop inside the fiber and one external regulator with $l = 1$. Therefore this building block is classified as $|n = 1, l = 1\rangle$. On the other hand, the input tree of genes *ung*, *tsr* and *psd* is finite with one external regulator, therefore this building block is classified as $|n = 0, l = 1\rangle$.

These considerations lead to a systematic classification of all building blocks. The results for the *E. coli* TRN are summarized in Fig. 14.9 and Fig. 14.10, showing how dissimilar circuits can be concisely classified by either integer dimensions $|n, l\rangle$ or fractal dimensions. The count for each building block is shown in Table 14.2. We explore this classification one circuit at a time in the next chapter.

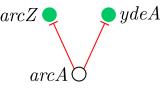
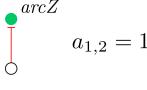
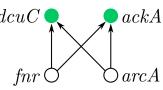
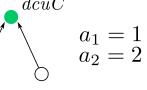
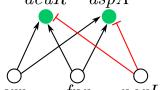
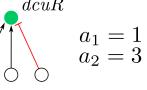
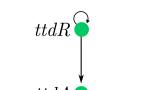
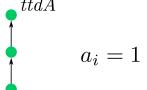
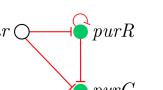
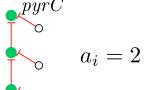
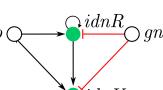
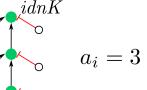
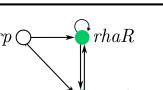
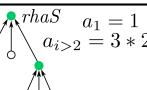
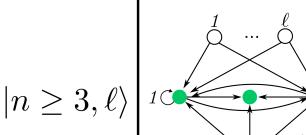
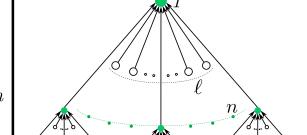
$ n, \ell\rangle$	Genetic circuit	Input tree	Base
$ 0, 1\rangle$			
$ 0, 2\rangle$			
$ 0, 3\rangle$			
$ 1, 0\rangle$			
$ 1, 1\rangle$			
$ 1, 2\rangle$			
$ 2, 1\rangle$			
$ n \geq 3, \ell\rangle$			

Fig. 14.9

Classification of simple fibration building blocks with integer dimensions. Basic fibration building blocks found in *E. coli*. These building blocks are characterized by a fiber that does not send back information to its regulator. They are characterized by two integer fiber numbers: $|n, \ell\rangle$. We show selected examples of circuits and input trees and bases. The statistical count of every class will be discussed later. The last example shows a generic building block for a general n -ary tree $|n, \ell\rangle$ with ℓ regulators. Figure reproduced from (Morone et al., 2020). Copyright © 2020, National Academy of Sciences U.S.A.

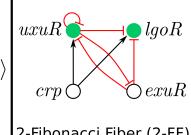
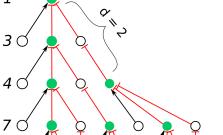
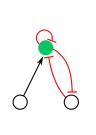
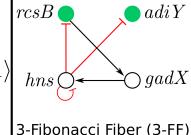
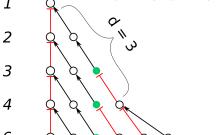
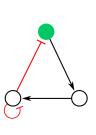
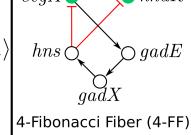
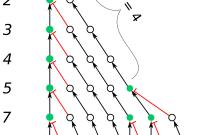
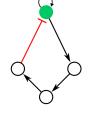
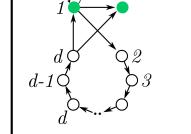
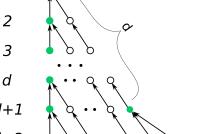
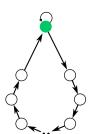
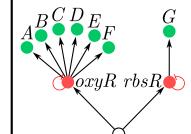
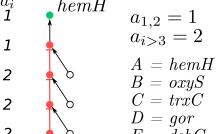
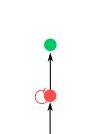
$ \varphi_d, \ell\rangle$	Genetic circuit	Input tree	Base
$ 1.6180..., 2\rangle$			
$ 1.4655..., 1\rangle$			
$ 1.3802..., 1\rangle$			
$ \varphi_d, \ell\rangle$			
$ 0, 1\rangle \oplus 1, 1\rangle$			

Fig. 14.10

Classification of simple building blocks with fractal dimensions (although more complex than Fig. 14.9). Fibonacci and multilayer building blocks. These building blocks are more complex than the simpler ones shown in Fig. 14.9. They are characterized by an autoregulated fiber that sends back information to its regulator. This creates a fractal input tree that encodes a Fibonacci sequence with golden branching ratio in the number of paths a_i . When the information is sent to the connected component that includes the regulator, then a cycle of length d is formed and the topology is a generalized Fibonacci block with golden ratio φ_d as indicated. Last panel shows a multilayer composite fiber with a feed-forward structure. Figure reproduced from (Morone et al., 2020). Copyright © 2020, National Academy of Sciences U.S.A.

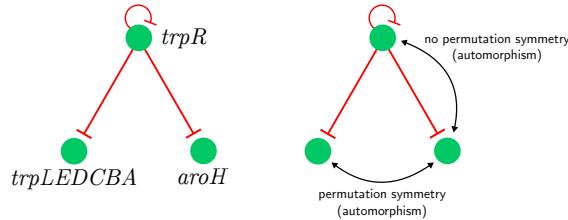
We use the hierarchy of fibration building blocks introduced in the previous chapter to analyze several small circuits that constitute such building blocks. They all occur widely in biology. We discuss their topology and investigate their synchronous dynamics using a variety of different models. We suggest that the resulting hierarchy of increasingly complex building blocks may have evolved through a sequence of changes in network topology, obtained by repeated application of the lifting property for fibrations of Section 6.5.

15.1 The primordial building block: autoregulation (AR) loop

The first nontrivial form of synchronization in the hierarchy of fibration building blocks occurs when a TF autonomously regulates its own expression, establishing an autoregulation (AR) loop, and subsequently controls the expression of other genes. This phenomenon is illustrated in Fig. 15.1a. This circuit is widely present in *E. coli*. One example is the tryptophan biosynthesis process, where it is regulated by TrpR (Grove and Gunsalus, 1987). TrpR represses its own expression, as well as the gene *aroH* (2-Dehydro-3-deoxyphosphoheptonate aldolase) and the *trpLEDCBA* operon, which encodes the enzymes involved in tryptophan biosynthesis. The activation of this circuit is dependent on the intracellular level of L-tryptophan (Karp et al., 2018). When the cell contains sufficient tryptophan, the TF binds to the operon and suppresses gene expression.

This circuit exhibits an automorphism which corresponds to permutation of the genes regulated by *trpR* as shown in Fig. 15.1a (right). It generates the symmetry group \mathbb{S}_2 . This is a rather trivial automorphism which appears many times in all regulatory networks. It is called a *regulon*. A regulon is a group of genes that are regulated as a unit by the same TF, and no other TF regulates any of the genes.

The interesting feature is in the AR of *trpR*. This AR loop at *trpR* does not affect the automorphism permuting the regulated genes. However, the AR loop introduces a fibration symmetry between *trpR*, *trpLEDCBA*, and *aroH*. This symmetry cannot be captured by a permutation of the nodes. For example, permuting the operon *trpLEDCBA* with *aroH* preserves adjacency, but permuting *trpR* with either the operon or *aroH* does not preserve adjacency. Therefore, *trpR* does not belong to the symmetry group acting on *trpLEDCBA* and *aroH*. However, *trpR* is synchronized with *trpLEDCBA* and *aroH* by the symmetry fibration, because they have isomorphic input trees, as shown in Fig. 15.1b (left). The base

a $|1, 0\rangle$: AR loop plus regulon

b Fibration symmetry in the input tree

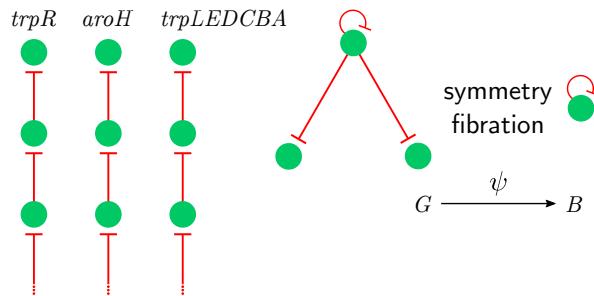


Fig. 15.1

Simple fibration building block 1: AR loop with regulon. (a) Genes *aroH* and *trpLEDCBA* can be permuted by the S_2 symmetry group, while *trpR* cannot be permuted with them without changing the network. (b) *trpR* receives an input only from itself, therefore its input tree is an infinite chain. *aroH* and *trpLEDCBA* receive an input from *trpR*, which in turn receives an input from itself, turning these input trees into chains too. Therefore the input trees of all 3 genes are isomorphic to each other. Thus *aroH*, *trpLEDCBA* and *trpR* belong to the same fiber and can synchronize their activity. The circuit has one loop and no external regulators, so it is classified as $|n = 1, l = 0\rangle$. Figure reproduced from (Leifer et al., 2021). Copyright © 2021, The Author(s).

of this building block is simple: an AR loop as in Fig. 15.1b (right). This is the simplest form of fibration, reducing the circuit to what could be considered as a primordial genetic circuit: a single gene regulating itself.

Due to the AR loop, the input trees of this fiber satisfy $a_i = 1$ for all i , so the branching ratio is $n = 1$. This circuit does not have any external regulators, so $l = 0$. We therefore refer to this circuit as $|1, 0\rangle$.

While n captures the loops, and l captures the regulators, the number of regulated genes is still unspecified. Considering that many building blocks contain regulons, we extend the definition of fiber numbers given in 14.3 to include the number m of regulated genes of the fibers. This lead to the following extended definition:

Definition 15.1 Extended fiber number classification of simple fibration building

blocks. Definition 14.3 is extended to include the number m of regulated genes in the fiber:

$$|n, l, m\rangle : \text{extended simple building block classification.} \quad (15.1)$$

The building block in Fig. 15.1 becomes $|n, l, m\rangle = |1, 0, 2\rangle$, which, by the application of the fibration, becomes $|1, 0\rangle$ when the 2-gene regulon is collapsed. Since these two circuits behave the same dynamically, we typically use the shorter notation for the building block and ignore m . While these are not a complete set of invariants for circuit topologies, they provide a useful coarse classification covering most circuits studied in this book.

The AR building block is quite abundant in simple bacterial genomes like the *E. coli* TRN as shown in the statistics in Table 14.2. However, AR loops are not present in more complex genomes like eukaryotes. The base of this circuit is also one of the network motifs found by Milo et al. (2002) in the same TRN. In fact, Milo et al. (2002) already found that AR loops are quite abundant in *E. coli*. However, the fiber is more complex than the AR motif in (Milo et al., 2002), since it contains not only the AR base but also the regulon. Thus, the fibration building block captures the correct synchrony of the genes, which cannot be captured by the AR-motif building block.

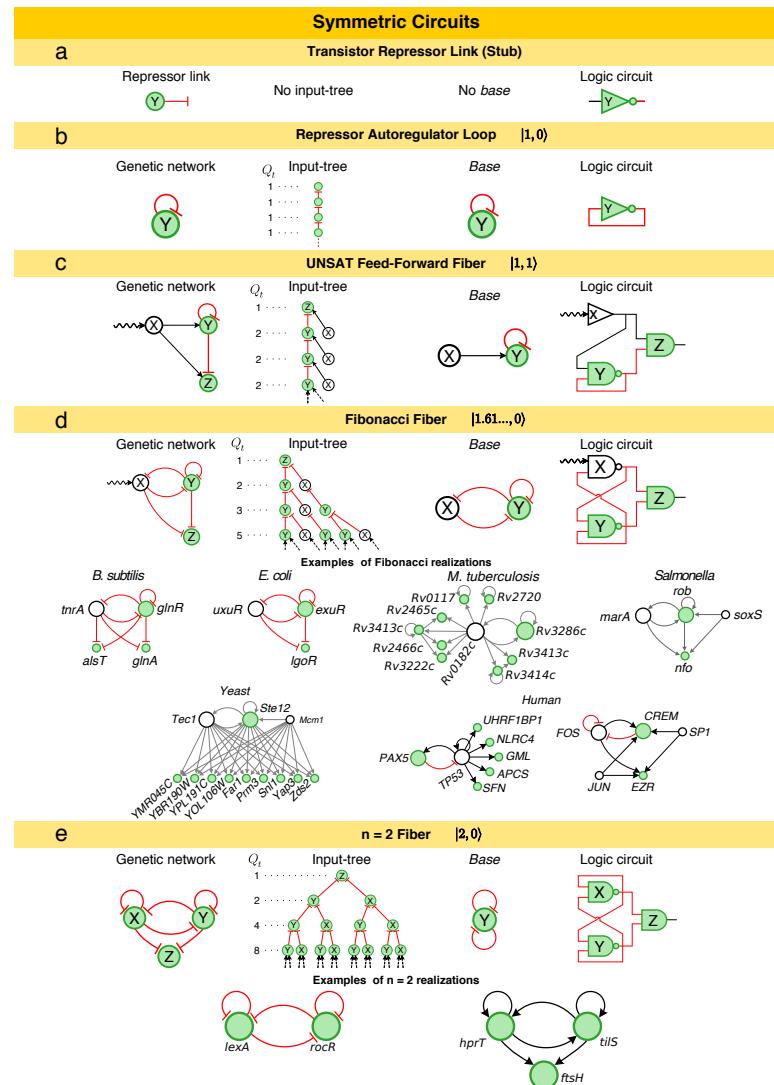
Building upon this primordial circuit, the rest of the building blocks appear in a well defined hierarchy of circuits, emerging from the lifting property of the fibration, see Section 6.5.

15.2 Dynamic repertoires in classes of fibration building blocks

The simplest possible base for a fibration consists of one autoregulated gene. By the lifting property (Definition 6.12) this base can be lifted to produce its regulon, conforming to the fiber observed in *E. coli*. We come back to this idea in subsequent chapters: the evolution of the building blocks into a hierarchical classification can be explained by the lifting property applied to simple bases like the AR loop. For now, we continue to describe the simple circuits found in *E. coli*.

The procedure to build more complex fibers can be systematically extended through an algebra of circuits that adds external regulators and loops to grow the base of symmetric circuits (Fig. 15.2). In this space, the AR is the core loop unit $|n = 1, l = 0\rangle$ (Fig. 15.2b).

From the AR base $|n = 1, l = 0\rangle$ we can increase the number of regulator genes, $|1, l\rangle$ with $l > 1$, as well as the number of regulated genes, by lifting. If this is done by following the lifting of the base, it does not affect the complexity (measured by the branching ratio n of the input tree) because all the relevant dynamics remain constrained to the sole loop in the fiber. This changes as soon as the fiber feeds information back to the external regulator, as for the circuits in Fig. 15.2d, where the gene Y now regulates its own regulator gene X. Addition of a second feedback loop results in an input tree that follows the Fibonacci sequence $a_t = 1, 2, 3, 5, 8, \dots$. Here, gene X is not part of the base. The branching structure

**Fig. 15.2**

Hierarchy of simple fibration building blocks. Black arrows indicate activator links, red denotes repressors, and gray denotes unknown functionality. Examples are shown from all studied species. Symmetric circuits function as clocks. Adding autoregulation and feedback loops results in a hierarchy of increasingly complex circuits. Turning the repressor link (**a**) into repressor autoregulation (**b**) gives an input tree equivalent to its own base. (**c**) Adding an external regulator creates the UNSAT-FFF, where genes Y and Z synchronize and oscillate. (**d**) Adding a second feedback loop gives an input tree that branches like the Fibonacci sequence. (**e**) A second autoregulation at X results in a symmetric input tree with branching ratio $n = 2$. This fiber collapses into a base with two autoregulators. Figure reproduced from (Leifer et al., 2020). Copyright © 2020, Leifer et al.

of the input tree implies that the Fibonacci fiber can oscillate and synchronize. A second autoregulation at X results in a symmetric input tree with $a_t = 2a_{t-1}$ and branching ratio $n = 2$. This fiber collapses into a base with two autoregulators. Examples of the $n = 2$ fiber are two fibers from the regulatory networks of *Bacillus subtilis*. The important component of these circuits is the delay in the feedback loop through the regulator from Y → X and back to Y. We show in Chapter 18 that these building blocks form the base of memory circuits that emerge via the lifting property and by symmetry breaking, forming memory devices that mimic the universal storage devices of computer memories.

Lifting of genes mimics the duplication mechanism in genetic evolution, see Fig. 18.2. Therefore, from the starting point of an AR base, we can create a hierarchy of circuits that mimics a biological evolutionary dynamic of growing circuits, which increases the number of regulated genes and adds edges in an orderly fashion. The simplest circuits involved are shown in Fig. 8.4 in Chapter 8, where we analyzed some features of their dynamics. Now we add biological and dynamical detail. The initial steps of this hierarchical evolution of circuits of increasing complexity are depicted in Fig. 15.3, and will be discussed next, circuit by circuit.

15.3 The feed-forward fiber - FFF

We have seen that the AR base can be extended by lifting the base to the regulon made of m regulated genes. This process does not change the dynamics of the resulting fiber since the regulon can be collapsed back to the base by the fibration without changing its dynamics. The lifting can be done to any number m of genes, increasing the complexity from $m = 0$ at the base to m genes in the fiber. Beyond adding regulated genes, we can imagine that another way to increase the complexity of the AR base is to add regulators by increasing $l = 0$ to $l = 1$. When we also add an $m = 1$ regulated gene we create a new building block that is quite abundant across all species and bio-networks. We call it the Feed-Forward Fiber (FFF) (Morone et al., 2020) denoted by $|1, 1\rangle$.

An example of an FFF is observed in the purine biosynthesis circuit in *E. coli*, Fig. 15.4a. It is composed of the repressor TF *purR* and its target gene *pyrC*, both regulated by the master regulator *fur*. The input trees of the genes in this FFF are shown in Fig. 15.4a. We see that *pyrC* and *purR* receive the same inputs from both *fur* and *purR*. On the first layer of the input tree, we find that *purR* is an autoregulator and also regulates the gene. The second level of the input tree contains exactly the same genes, and so on. The loop at *purR* creates an input tree with infinitely many layers.

The FFF resembles the feed-forward loop (FFL) network motif introduced by Milo et al. (2002), except for an additional AR at the intermediate TF *purR*. This crucial addition transforms an FFL into an FFF composed of two genes in the synchronized fiber. This type of building block is abundant in bacterial TRN (Leifer et al., 2020), see Table 14.2.

The dynamics of the $|n = 1, l = 1\rangle$ FFF building block has been studied in detail in (Leifer et al., 2020). The FFF has a synchronous oscillatory solution for a certain type of regulation, which allows it to play the role of a clock in the system. Therefore, the FFF is of

Class	FFF		Fibonacci	$n = 2$		
Sub-class	0–FFF	1–FFF	φ –FFF	2–FFF		
Base						
<i>E. coli</i> Realization				lock-on 	toggle-switch 	Smolen
Other Realizations				lock-on 	toggle-switch 	Smolen

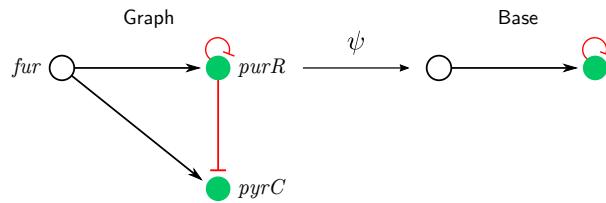
Fig. 15.3

Evolution of simple fibration building blocks by lifting (duplication) and addition of edges. Realizations of fibration building blocks and their broken symmetry version. The consideration of external regulator nodes, feedback loops, and autoregulation loops results in circuits with an increasing level of complexity. *First column of graphs:* FFF (feed-forward fiber). The base for the 0-FFF and (1-FFF), namely, AR (autoregulation) loop and the AR loop with an external regulation unfold (or ‘lift’) into different circuits found in the *E. coli* GRN, as the FFF, or synthetic ones, as the Repressilator. *Second column: Fibonacci.* The Fibonacci (φ -FF) base results in the Fibonacci circuit by unfolding the AR loop into two synchronized genes. An example present in the GRN of *E. coli* is the circuit made by the *Uxur*, *Exur*, and *Lgor* genes. *Third column: n=2.* The presence of the second regulation loop allows diversity in combinations between gene regulations, resulting in a different subclass $n = 2$ with different circuits and dynamics. The realizations presented are both symmetric and broken symmetric ones found in *E. coli* from (Leifer et al., 2020), which are related to previous GRNs found in the literature (lock-on, toggle-switch, and Smolen). Figure reproduced from (Stewart et al., 2024). Copyright © 2024, The Author(s).

high interest for information processing networks. We now compare it with the analogous FFL motif.

15.3.1 Feed-forward loop (FFL) network motif

We consider the dynamics of the most abundant network motif in gene regulatory networks, the feed-forward loop (FFL) (Shen-Orr et al., 2002; Mangan and Alon, 2003; Mangan et al.,

a $|1, 1\rangle$: AR loop plus regulon

b Input trees

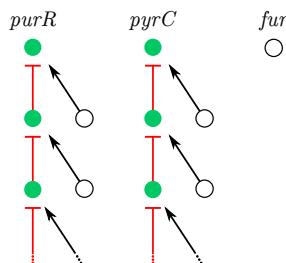


Fig. 15.4

Simple fibration building block 2: Feed-forward fiber (FFF). It is composed of an AR loop with a regulon and external regulator. (a) *purR* and its target gene *pyrC* regulated by *fur* form a FFF. FFF has one loop and one external regulator and is classified as $|n = 1, l = 1\rangle$. *purR* and *pyrC* belong to the same fiber (shown in (b)) and are collapsed by the fibration ψ , while *fur* is left untouched. (b) *purR* receives an input from itself, creating an infinite chain, together with regulator *fur*, which does not have any inputs. Therefore the input tree of *purR* is an infinite chain with an additional input on each layer. Similarly, *pyrC* receives an input from *purR* that leads to an infinite chain, and *fur* creates an additional input. *fur* does not receive any inputs, and therefore has an input tree of height 0. Input trees of *purR* and *pyrC* are isomorphic, therefore *purR* and *pyrC* belong to the same fiber and synchronize their activity. Figure reproduced from (Leifer et al., 2021). Copyright © 2021, The Author(s).

2003). In particular the coherent cFFL motif consists of three activator genes X, Y, and Z, where the transcription factor expressed by gene X positively regulates the transcription of Y and Z, and in turn Y regulates Z (Fig. 15.5a). Figure 15.5a shows an example of the cFFL motif in *E. coli* with X=*cpxR*, Y=*baeR*, and Z=*spy*. Numerical and analytic solutions for the expression levels of the genes in the cFFL (Fig. 15.5b) demonstrate that the FFL cannot attain synchronization (unless for specific *ad hoc* settings of parameters) nor oscillations in expression levels. This is consistent with previous research which has interpreted the functionality of the FFL as a sign-sensitive signal delay in transcriptional networks (Mangan et al., 2003).

Leifer et al. (2020) illustrate this result by presenting an analytic solution of the FFL using a simple model of gene expression dynamics that uses a discrete-time, continuous state

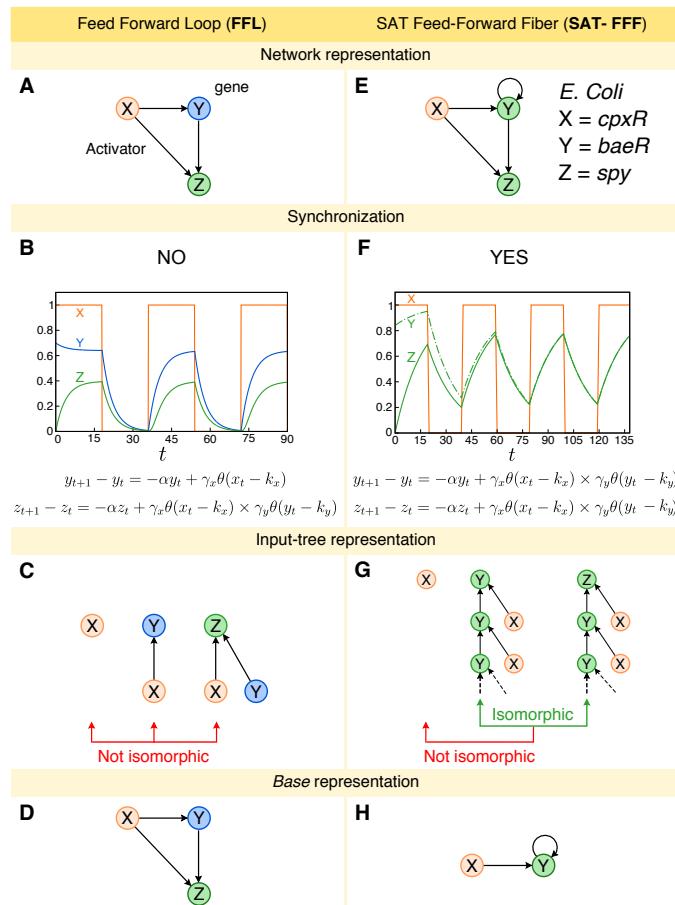


Fig. 15.5

Comparison between the Feed-Forward Loop (FFL) and feed-forward fiber (FFF).

(a) FFL network representation. (b) Numerical solution of FFL dynamics. The expression levels of genes Y and Z do not synchronize. The oscillation pattern presented is due to the square-wave behavior of gene X expression levels. (c) Input trees of FFL. The input trees of genes X, Y, and Z are not isomorphic, so their expression levels do not synchronize. (d) Base representation of FFL. The base is the same as the original circuits since there are no symmetries. (e) SAT-FFF network representation. Addition of autoregulation leads to a symmetry between the expression levels of genes Y and Z. (f) The numerical solution of the SAT-FFF dynamics shows synchronization of the expression levels of genes Y and Z. Again, the oscillation is due to the wave-like pattern of X. (g) Genes Y and Z have isomorphic input trees and synchronize. The input tree of the external regulator X is not isomorphic to these, even though it directly regulates the fiber. (h) Since Y and Z synchronize, gene Z can be collapsed into Y, resulting in a simpler base representation. Figure reproduced from (Leifer et al., 2020). Copyright © 2020, Leifer et al.

variable model with a logic Boolean interaction function in the spirit of the Glass–Kauffman model of biochemical networks (Glass and Kauffman, 1973). Alon (2019), Mangan and Alon (2003) and Mangan et al. (2003) studied the dynamics of the expression levels y_t and z_t of genes Y and Z, respectively, as a function of time t in the cFFL. The discrete dynamical system is given by the following difference equations:

$$\begin{aligned} y_{t+1} &= (1 - \alpha)y_t + \gamma_x \theta(x_t - k_x), \\ z_{t+1} &= (1 - \alpha)z_t + \gamma_x \theta(x_t - k_x) \gamma_y \theta(y_t - k_y) \end{aligned} \quad (15.2)$$

where x_t is the expression level of gene X, α is the degradation rate of the gene, γ_x and γ_y are the strength of the interaction representing the maximum expression rate of genes X and Y, respectively, and the thresholds k_x and k_y are the dissociation constant between the transcription factor and biding site. The expression level is measured in terms of abundance of gene product, e.g., mRNA concentration. The Heaviside step functions $\theta(x_t - k_x)$ and $\theta(y_t - k_y)$ represent the activator regulation from gene X and Y, respectively. They represent the Boolean logic approximation of Hill input functions in the limit of strong cooperativity (Alon, 2019). We consider an AND gate for the combined interaction of transcription factors of genes X and Y onto the binding sites of gene Z. Leifer et al. (2020) show that analogous results can be obtained with an OR gate and with ODE continuum models. Solutions for the FFL have been considered in the literature (Alon, 2019). Here, we adapt those results to the particular models used in our studies to perform consistent comparisons with the solutions of the FFF discussed next.

Leifer et al. (2020) showed that the expression levels of the genes Y and Z do not synchronize; that is, y_t and z_t are not the same for all t . This behavior is exemplified in Fig. 15.5b, which shows the solution for a particular set of parameters resulting in a non-synchronized state. Such a state is obtained under the initial conditions $y_0 > k_y$ and $\alpha < \gamma_x$. Specifically, we use the following parameters: $\alpha = 0.2$, $\gamma_x = 0.12$, $\gamma_y = 0.7$, $k_x = 0.5$ and $k_y = 0.1$. For this combination, y_t and z_t do not synchronize since y_t saturates at $y_t \rightarrow \gamma_x/\alpha = 0.6$ when $t \rightarrow \infty$, and z_t saturates at $z_t \rightarrow \gamma_x \gamma_y / \alpha = 0.42$, for $t \rightarrow \infty$.

In the figure, we set x_t equal to a square wave and then monitor the expression levels of y_t and z_t . When $x < k_x$, both y_t and z_t decay exponentially to zero. On the other hand, when $x > k_x$, both variables evolve to saturate again at $y_t = \gamma_x/\alpha$ and $z_t = \gamma_x \gamma_y / \alpha$, in agreement with the analytical solution.

These results are consistent with a continuous variable approach (Alon, 2019; Mangan and Alon, 2003; Mangan et al., 2003) using ODEs governing the dynamics of expression levels $y(t)$ and $z(t)$:

$$\begin{aligned} \dot{y}(t) &= -\alpha y(t) + \gamma_x \theta(x(t) - k_x), \\ \dot{z}(t) &= -\alpha z(t) + \gamma_x \theta(x(t) - k_x) \gamma_y \theta(y(t) - k_y). \end{aligned} \quad (15.3)$$

This model contains many assumptions about the values of the different parameters. For instance, α is the same for both genes, and so is γ_x . In Section 21.3.1 we refine this idealistic model of gene expression to discuss how these approximations can be broken in a more realistic model.

This model shows that the expression levels from genes Y and X do not synchronize. In

addition, $y(t)$ and $z(t)$ also do not reach oscillatory states, in accordance with the results of the discrete time model.

15.3.2 Feed-forward fiber synchronization

Fibers can predict which genes can synchronize, but they cannot predict what kinds of synchronized states—e.g. fixed point synchronization or limit cycle oscillations—these genes may reach. To determine the type of dynamical state achieved by the fiber, we need to analyze its dynamics using systems of ODEs. In general, more complex input tree structures allow circuits to display additional dynamical states. For instance, while a FFF fiber with all activators leads to a simple fixed point, repressors and cycles in the fiber can give rise to oscillatory limit cycles. Even a minimal loop, such as the AR in an FFF, can produce oscillatory dynamics if the regulatory interactions contain a delay (Leifer et al., 2020). In reality, translation and transcription might create such delays. Longer cycles, like those observed in the Fibonacci fibers, allow for more complex functionality beyond the simple FFF regulation of a single input function and can yield oscillations with varying periods and more stability across the parameter space, which extends the signal processing capabilities of the circuits. We systematically describe the dynamics of the simple building blocks, starting with the FFF.

The lack of synchronization in the FFL is easily explained by the lack of symmetry in this circuit: gene Z receives input from X and Y, while gene Y, instead, only from X, and therefore the inputs are not symmetric (Fig. 15.5c). However, a search of motifs in biological networks show that the FFL circuit does not appear alone in the bacterial TRN: instead, it often appears in conjunction with the AR loop (like $Y=baeR$ in Fig. 15.5e) forming a synchronized FFF. Because network motif algorithms search for these two circuits separately, they were never found together, even though they appear together, forming the FFF in the bacterial TRN. This exemplifies the problem with network motifs or any searching algorithm based on statistics alone: they have difficulties finding biologically meaningful circuits because statistical significance does not guarantee biological functionality.

Numerical simulations in Fig. 15.5f and analytical solutions of the FFF done by Leifer et al. (2020) confirm the synchronization of genes Y and Z into coherent coexpression, as predicted by the fibration.

The FFF with activator regulation in Fig. 15.5e has a simple dynamic converging to a synchronized fixed point: all interactions are satisfied, so the Heaviside step functions evaluate to 1. We call this circuit SAT-FFF since its interactions are always *satisfied* and positive. Instead, when the autoregulation is a repressor, the loop behaves as a logical NOT gate. When expression is high, it inhibits itself, shifting to a low state. Dually, if it is low, then it promotes itself to shift to a high state. Hence, the activity of this gene oscillates indefinitely (Atkinson et al., 2003; Glass and Kauffman, 1973). We call this circuit the UNSAT-FFF since the interactions are *unsatisfied*. This is the simplest expression of *frustration* (Anderson, 1972; Glass and Kauffman, 1973), a core concept in complexity and spin glasses, which refers to a system that is always in tension and thus never reaches a stable fixed configuration.

15.3.3 Satisfied feed-forward fiber (SAT-FFF) synchronization

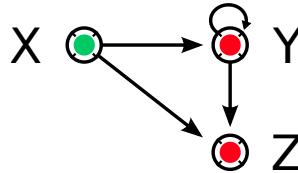


Fig. 15.6

SAT-FFF network representation. SAT-FFF is an FFF in which all links between the nodes in the fiber are activation links, therefore, they are satisfied.

The SAT-FFF is a feed-forward fiber with activator autoregulation where all interactions are satisfied. That is, it does not present the phenomenon of frustration, and the dynamics converge to a fixed point. This can easily be seen by considering gene X to be high, which makes genes Y and Z high, too. Finally, the configuration satisfies the AR loop, so all bonds are satisfied. Below, we describe the solutions of the SAT-FFF circuit in a discrete-time continuous variable model and ODE model.

A. SAT-FFF discrete time model

The discrete-time dynamics of the SAT-FFF with a logic interaction term is given by:

$$\begin{aligned} y_{t+1} &= (1 - \alpha)y_t + \gamma_x \theta(x_t - k_x) \gamma_y \theta(y_t - k_y), \\ z_{t+1} &= (1 - \alpha)z_t + \gamma_x \theta(x_t - k_x) \gamma_y \theta(y_t - k_y). \end{aligned} \quad (15.4)$$

The Heaviside function $\theta(y_t - k_y)$ represents activator feedback on the autoregulation of the Y gene. We consider an AND gate for the interactions (Alon, 2019) (analogous results can be obtained for OR gates). Writing down the set of equations for the rescaled variables $\psi_t = y_t/k_y$ and $\zeta_t = z_t/k_y$, we get:

$$\begin{aligned} \psi_{t+1} &= \beta\psi_t + \alpha\lambda\theta(x_t - k_x)\theta(\psi_t - 1), \\ \zeta_{t+1} &= \beta\zeta_t + \alpha\lambda\theta(x_t - k_x)\theta(\psi_t - 1). \end{aligned} \quad (15.5)$$

Here $\lambda = \gamma_x\gamma_y/\alpha k_y$ and $\beta = (1 - \alpha)$. Since the second terms on the right-hand sides of both equations are equal, the dynamical variables ψ_t and ζ_t must synchronize, as well as y_t and z_t . Again, considering $x_t = x$ constant, for $x < k_x$, the solutions for ψ_t and ζ_t are trivial: both variables decay exponentially as $\psi_t = \psi_0 e^{-t/\tau}$ and $\zeta_t = \zeta_0 e^{-t/\tau}$, where $\tau^{-1} = -\log(1 - \alpha)$. This behavior is shown by the red solid line in Fig. 15.7b with $\psi_0 = 0.9$.

In terms of the iterative map, the dynamics of the SAT-FFF for the rescaled variable ψ_t with $x > k_x$ is:

$$\psi_{t+1} = \beta\psi_t + \alpha\lambda\theta(\psi_t - 1) \equiv f(\psi_t), \quad (15.6)$$

so we find

$$f^t(\psi) = f^{t-1}(\beta\psi)\theta(1 - \psi) + f^{t-1}(\beta\psi + \lambda)\theta(\psi - 1). \quad (15.7)$$

This iterative map $\psi_t = f(\psi_t)$ provides different solutions depending on ψ_0 . Similarly to the case $x < k_x$, if $\psi_0 < 1$ the solution decays to zero as $\psi_t = \psi_0 e^{-t/\tau}$. However, if $\psi_0 > 1$, there are two possibilities, depending on the values of $\lambda = \gamma_x \gamma_y / \alpha k_y$.

First, if $\lambda > 1$, the solution for both rescaled variables converges to λ as $\psi_t = \psi_0 e^{-t/\tau} + \lambda(1 - e^{-t/\tau})$ and $\zeta_t = \zeta_0 e^{-t/\tau} + \lambda(1 - e^{-t/\tau})$, such that $\psi_{t \rightarrow \infty} \rightarrow \lambda$ and $\zeta_{t \rightarrow \infty} \rightarrow \lambda$, represented by the blue dash-dotted line in Fig. 15.7b. For this case, we use $\psi_0 = 1.1$ and $\lambda = 2$.

For $\lambda < 1$, ψ_t approaches 1 at a time t^* given by

$$t^* = \left\lceil \frac{1}{\log(1 - \alpha)} \log \left(\frac{1 - \lambda}{\psi_0 - \lambda} \right) \right\rceil. \quad (15.8)$$

For $t > t^*$ the solutions decay to zero as $\psi_t = e^{-(t-t^*)/\tau}$ and $\zeta_t = \zeta_{t^*} e^{1(t-t^*)/\tau}$. This behavior is represented in Fig. 15.7b by the dashed green line, where we use $\psi_0 = 2$ and $\lambda = 0.9$. The rescaled variables ψ_t and ζ_t always synchronize, and so do y_t and z_t . This can be proved by finding the difference $\varepsilon_t = \psi_t - \zeta_t$. For all the cases discussed above, ε_t decays exponentially fast as $\varepsilon_t = (\psi_0 - \zeta_0) e^{-t/\tau}$.

We can use the solution with x_t constant to qualitatively understand the SAT-FFF in general. An example of a SAT-FFF with non-constant x_t is depicted in Fig. 15.5. As shown, the variables y_t and z_t synchronize, but with no internal oscillations. We feed an external oscillatory pattern of x_t as a square wave. For $x_t < k_x$, both y_t and z_t decay exponentially. When $x_t > k_x$, they tend to saturate at $\gamma_x \gamma_y / \alpha$. The SAT-FFF synchronizes at a fixed point.

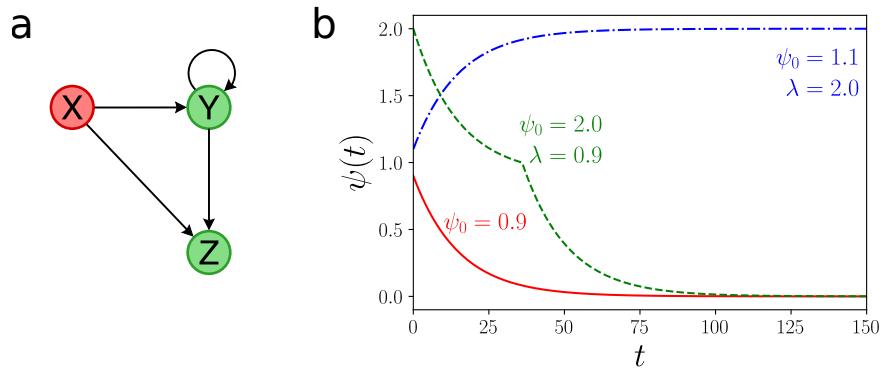


Fig. 15.7

SAT-FFF. (a) Network representation. (b) Different behaviors for the analytical solutions of ψ_t depending on ψ_0 and λ . Figure reproduced from (Leifer et al., 2020). Copyright © 2020, Leifer et al.

B. SAT-FFF ODE model

Now we consider the ODE model of SAT-FFF to confirm results from the discrete time continuous variable model in (Leifer et al., 2020). The dynamics of gene X is driven by outside sources, so we consider only the dynamics of genes Y and Z, which are described by:

$$\begin{aligned}\dot{y} &= -\alpha y(t) + \gamma_x \theta(x(t) - k_x) \gamma_y \theta(y(t) - k_y), \\ \dot{z} &= -\alpha z(t) + \gamma_x \theta(x(t) - k_x) \gamma_y \theta(y(t) - k_y).\end{aligned}\quad (15.9)$$

Taking $\psi(t) = y(t)/k_y$, $\zeta(t) = z(t)/k_y$ and $\delta = \gamma_x \gamma_y / k_y$ we transform (15.9) to:

$$\begin{aligned}\dot{\psi} &= -\alpha\psi(t) + \delta \theta(x(t) - k_x) \theta(\psi(t) - 1), \\ \dot{\zeta} &= -\alpha\zeta(t) + \delta \theta(x(t) - k_x) \theta(\psi(t) - 1).\end{aligned}\quad (15.10)$$

Without loss of generality, consider the case when $x(t) = x$ is constant over time. If $x < k_x$ then the solution of (15.10) is:

$$\begin{aligned}\psi(t)_{x < k_x} &= \psi_0 e^{-\alpha t}, \\ \zeta(t)_{x < k_x} &= \zeta_0 e^{-\alpha t},\end{aligned}\quad (15.11)$$

where ψ_0 and ζ_0 are the initial conditions.

Consider the case $x > k_x$. Now (15.10) transforms into:

$$\begin{aligned}\dot{\psi} &= -\alpha\psi(t) + \delta \theta(\psi(t) - 1), \\ \dot{\zeta} &= -\alpha\zeta(t) + \delta \theta(\psi(t) - 1).\end{aligned}\quad (15.12)$$

Because Y and Z belong to the same fiber, $\psi(t)$ and $\zeta(t)$ synchronize and therefore $y(t)$ and $z(t)$ also synchronize, so we may consider only the dynamics of the first equation:

$$\dot{\psi} = -\alpha\psi(t) + \delta \theta(\psi(t) - 1). \quad (15.13)$$

It is easy to see that for $\psi_0 < 1$, (15.11) is the solution of (15.13). When $\psi_0 > 1$ and $\delta/\alpha > 1$, the solution is:

$$\psi(t) = \delta/\alpha + (\psi_0 - \delta/\alpha)e^{-\alpha t}. \quad (15.14)$$

In the case $\psi_0 > 1$ and $\delta/\alpha < 1$, the dynamics of ψ is split into two parts: one before ψ decays to 1, and the other one after ψ crosses 1. The time when $\psi(t)$ crosses 1 is:

$$t_c = \frac{1}{\alpha} \ln\left(\frac{\psi_0 - \delta/\alpha}{1 - \delta/\alpha}\right), \quad (15.15)$$

and the dynamics can be written as:

$$\begin{aligned}\psi(t) &= \delta/\alpha + (\psi_0 - \delta/\alpha)e^{-\alpha t} && \text{for } t \in [0, t_c], \\ \psi(t) &= \frac{\psi_0 - \delta/\alpha}{1 - \delta/\alpha} e^{-\alpha t} && \text{for } t \in [t_c, \infty].\end{aligned}\quad (15.16)$$

To summarize, the solution is:

$$\begin{aligned}
 \psi_0 < 1 &\rightarrow \psi(t) = \psi_0 e^{-\alpha t} \\
 \psi_0 > 1, \frac{\delta}{\alpha} > 1 &\rightarrow \psi(t) = \delta/\alpha + (\psi_0 - \delta/\alpha)e^{-\alpha t} \\
 \psi_0 > 1, \frac{\delta}{\alpha} < 1 &\rightarrow \psi(t) = \frac{\psi_0 - \delta/\alpha}{1 - \delta/\alpha} e^{-\alpha t} \quad \text{for } t \in [0, t_c] \\
 &\qquad\qquad\qquad \psi(t) = \delta/\alpha + (\psi_0 - \delta/\alpha)e^{-\alpha t} \quad \text{for } t \in [t_c, \infty].
 \end{aligned} \tag{15.17}$$

This solution is analogous to the one obtained for the discrete time model above. That is, dynamics of the SAT-FFF is described by a synchronous fixed point in the ODE model as well.

15.3.4 Unsatisfied feed-forward fiber oscillates and synchronizes

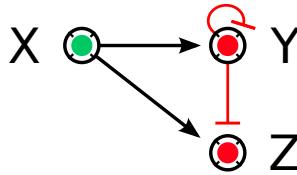


Fig. 15.8

UNSAT-FFF network representation. UNSAT-FFF is an FFF in which all links between the nodes in the fiber are repression links.

Leifer et al. (2020) show analytically and numerically that the UNSAT-FFF circuit has an oscillatory solution plus synchronization of genes Y and Z (as expected from the fibration) using a discrete-time continuous variable model and an ODE with time delay model. The time delay is related to the process of transcription and translation. The conclusions made for SAT-FFF and FFL hold in the framework of DDEs (Delay Differential Equations) but were considered without the delay for simplicity.

A. UNSAT-FFF discrete time model

The discrete-time dynamics of the expression levels of genes y_t and z_t in the UNSAT-FFF are given by:

$$\begin{aligned}
 y_{t+1} &= (1 - \alpha)y_t + \gamma_x \theta(x_t - k_x) \gamma_y \theta(k_y - y_t), \\
 z_{t+1} &= (1 - \alpha)z_t + \gamma_x \theta(x_t - k_x) \gamma_y \theta(k_y - y_t),
 \end{aligned} \tag{15.18}$$

where γ_x and γ_y are the strengths of the interaction (maximum expression rate) of genes X and Y, respectively, and k_x and k_y are the respective dissociation constants of the same

genes. Similarly to the SAT-FFF case, synchronization between y and z occurs because of the symmetry fibration that collapses nodes Y and Z . However, the impact of the repressor feedback loop on the dynamical behavior of this circuit is more profound, since it leads to oscillations. Thus, while both SAT-FFF and UNSAT-FFF lead to synchronization of Y and Z , the former synchronizes into a fixed point and the latter into an oscillatory limit cycle.

We set $\lambda = \gamma_x \gamma_y / k_y \alpha$, $\beta = 1 - \alpha$, and rewrite (15.18) for the rescaled variables $\psi_t = y_t / k_y$ and $\zeta_t = z_t / k_y$ as:

$$\begin{aligned}\psi_{t+1} &= \beta\psi_t + \alpha\lambda\theta(x_t - k_x)\theta(k_y - \psi_t), \\ \zeta_{t+1} &= \beta\zeta_t + \alpha\lambda\theta(x_t - k_x)\theta(k_y - \psi_t).\end{aligned}\quad (15.19)$$

We set $x_t = x$ constant in time for simplicity. For $x < k_x$, the solutions decay exponentially as $\psi_t = \psi_0 e^{-t/\tau}$ and $\zeta_t = \zeta_0 e^{-t/\tau}$, where ψ_0 is the initial condition. For $x > k_x$, (15.19) defines an iterative map which satisfies the following recursive equation:

$$f^t(\psi) = f^{t-1}(\beta\psi)\theta(\psi - 1) + f^{t-1}(\beta\psi + \alpha\lambda)\theta(1 - \psi). \quad (15.20)$$

This iterative map results in different solutions depending on the value of λ . We first consider the case where the initial condition is $\psi_0 > 1$. The solution of (15.19) is then $\psi_t = \psi_0 e^{-t/\tau}$, where $\tau^{-1} = -\log(1 - \alpha)$. This solution is correct as long as $\psi_t > 1$, but ceases to be valid at a certain time t^* such that $\psi_t < 1$, which is given by $t^* = \lceil \tau \log \psi_0 \rceil$.

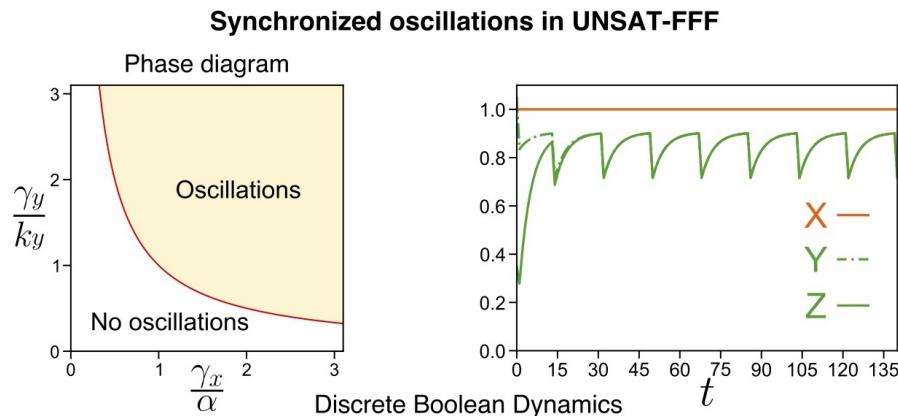


Fig. 15.9

Phase diagram of oscillations of the UNSAT-FFF. An oscillatory phase is defined by the condition $\gamma_y/k_y > (\gamma_x/\alpha)^{-1}$. For instance, on the right side, we plot the solution of (15.19) for a set of parameters satisfying such condition. Specifically, $\alpha = 0.205$, $\gamma_x = 0.454$, $\gamma_y = 0.454$, $k_x = 0.5$, and $k_y = 1.0$. Figure reproduced from (Leifer et al., 2020). Copyright © 2020, Leifer et al.

Next, we consider the case $\psi_0 < 1$. In this case the solution is given by $\psi_t = \psi_0 e^{-t/\tau} + \lambda(1 - e^{-t/\tau})$, which is always valid for $\lambda < 1$. Thus when $\lambda < 1$ the system does not oscillate, but converges monotonically to a fixed point $\psi_\infty = \lambda$. However, when $\lambda > 1$ this solution ceases to be valid at the time $t^* = \lceil \tau \log \frac{\lambda - \psi_0}{\lambda - 1} \rceil$ such that $\psi_t > 1$. Therefore the

solution ψ_t oscillates in time for $\lambda > 1$. For the case of $\psi_0 > 1$, the explicit solution is given by the following general analytic expression, which is plotted in Fig. 15.9 (right):

$$\begin{aligned}\psi_t &= \psi_0 e^{-t/\tau} && \text{for } t \in \{0, 1, \dots, t_1\}, \\ \psi_t &= \psi_1 e^{-(t-t_1)/\tau} + \lambda(1 - e^{-(t-t_1)/\tau}) && \text{for } t \in \{t_1, \dots, t_2\}, \\ \psi_t &= \psi_2 e^{-(t-t_2)/\tau} && \text{for } t \in \{t_2, \dots, t_3\}.\end{aligned}\quad (15.21)$$

Here

$$t_1 = \lceil \tau \log \psi_0 \rceil \quad t_2 = t_1 + \left\lceil \tau \log \frac{\lambda - \psi_1}{\lambda - 1} \right\rceil \quad t_3 = t_2 + \lceil \tau \log \psi_2 \rceil$$

The general solution with initial condition $\psi(t_0) < 1$ can be written in a similar way.

Thus the main condition for oscillations in the circuits is $\lambda > 1$. If $\lambda < 1$ there is no oscillatory behavior and the solution ψ_t converges monotonically to λ . Therefore, the oscillatory phase is separated from the non-oscillatory phase by the condition:

$$\frac{\gamma_y}{k_y} = \left(\frac{\gamma_x}{\alpha} \right)^{-1}, \quad (15.22)$$

which is depicted in the phase diagram of Fig. 15.9 (left).

B. UNSAT-FFF DDE model

Now we elaborate on the solution of the ODE continuum model. Since both genes Y and Z synchronize their behavior, the UNSAT-FFF circuit can be reduced to study the base of the circuit consisting of a negative autorregulation loop and an external regulator X (Fig. 15.2c, right). This circuit has been synthetically implemented by Stricker et al. (2008) using a promoter that drives the expression in the absence of LacI (and acts as a negative feedback loop) or in the presence of IPTG, which acts as an activator. It was shown experimentally that this circuit leads to oscillatory behavior in the expression profiles. This result was corroborated with a dynamical ODE model in (Stricker et al., 2008) which we adapt to study the case of the UNSAT-FFF with ODE. See also the review paper (Purcell et al., 2010).

As before, we consider gene $x(t) = x$ constant in time and larger than $x > k_x$, and rescale the expression of genes $y(t)$ and $z(t)$ as $\psi(t) = y(t)/k_y$ and $\zeta(t) = z(t)/k_z$. Since genes $y(t)$ and $z(t)$ synchronize their activities, then only one equation need be considered, that for $\psi(t)$.

The key to observing oscillations in a first-order ODE is to consider the delay in the signal propagation in the circuit. Without the delay, the dynamics converge to a fixed point; no oscillatory solution exists in a first-order ODE continuous time model on \mathbb{R} . The situation is different in the discrete-time model considered earlier. In this case, discrete-time plus a logic approximation leads to oscillations.

Negative feedback loop circuits with delays have been widely investigated in the dynamical systems literature. Here, we adapt the negative feedback loop model with a delay of (Stricker et al., 2008, Supplementary Information Equation (6)). We consider delays in the

negative feedback loop, which is the key feature explaining the experimentally observed robust oscillations in this circuit (Stricker et al., 2008).

Delays in a biological circuit arise from the combined processes of intermediate steps like transcription, translation, folding, multimerization and binding to DNA. This series of biological processes is lumped into a single arrow between two genes in the network representation of the circuit. In reality, this arrow represents processes that should be modeled in a more detailed manner. These biological processes can be approximately taken into account by inserting a delay in the interaction term in the dynamical equations. The interaction term can be written as $\delta \theta(1 - \psi(t - \tau))$, where τ represents the delay caused by the process of self-repression not being instantaneous. Therefore the dynamics of $\psi(t)$ is described by the first-order delay-differential equation (DDE) :

$$\dot{\psi} = -\alpha\psi(t) + \delta \theta(1 - \psi(t - \tau)), \quad (15.23)$$

where τ represents the delay caused by the expression process.

We find analytical solutions to this equation following a procedure outlined in (Driver, 2012, Chapter V). Initial conditions used for a DDE are not specified by the value of the function at one point, but rather by a set of values of the function on an interval of length τ . The solution of a DDE cannot be thought of as a sequence of values of $\psi(t)$ as in an ODE, but rather as a set of functions $\{f_0(t), f_1(t), f_2(t), \dots\}$, defined over a set of contiguous time intervals $\{[-\tau, 0], [0, \tau], [\tau, 2\tau], \dots\}$.

Consider (15.23) with initial function $f_0(t)$ for $t \in [-\tau, 0]$. Then for $t \in [0, \tau]$, (15.23) looks like:

$$\dot{\psi} = -\alpha\psi(t) + \delta \theta(1 - f_0(t - \tau)). \quad (15.24)$$

Moving the degradation term to the left and multiplying by $e^{\alpha t}$ we get:

$$\dot{\psi} e^{\alpha t} + \alpha\psi(t)e^{\alpha t} = \delta e^{\alpha t} \theta(1 - f_0(t - \tau)). \quad (15.25)$$

Rewriting the left part, we obtain:

$$\frac{d(\psi e^{\alpha t})}{dt} = \delta e^{\alpha t} \theta(1 - f_0(t - \tau)), \quad (15.26)$$

and integrating on the interval \int_0^t , we get:

$$\psi e^{\alpha t} - \psi(0) = \delta \int_0^t e^{\alpha t'} \theta(1 - f_0(t' - \tau)) dt'. \quad (15.27)$$

Since ψ is continuous at 0 ($\psi(0) = f_0(0)$) and $\psi(t)$ for $t \in [0, \tau]$ is given by $f_1(t)$, we write:

$$f_1(t) = f_0(0)e^{-\alpha t} + \delta \int_0^t e^{\alpha(t'-t)} \theta(1 - f_0(t' - \tau)) dt. \quad (15.28)$$

Following the same procedure we can derive the general formula for the solution $\psi(t)$ on the interval $[k\tau, (k+1)\tau]$, assuming that the solution on the previous interval $[(k-1)\tau, k\tau]$ is given by $f_{k-1}(t)$. The solution is then given by the following iterative equation:

$$\dot{\psi} = -\alpha\psi(t) + \delta \theta(1 - f_{k-1}(t - \tau)). \quad (15.29)$$

Applying the integrating factor method and integrating over $\int_{k\tau}^t$ we obtain:

$$\psi(t) = \psi(k\tau) * e^{\alpha(k\tau-t)} + \delta \int_{k\tau}^t e^{\alpha(t'-t)} \theta(1 - f_{k-1}(t' - \tau)) dt'. \quad (15.30)$$

Using (15.30) we recursively find functions $\{f_0(t), f_1(t), f_2(t), \dots\}$ on the interval of interest, which provide the solution to (15.23). Using Mathematica we find functions on the interval $t \in [-\tau, 30\tau]$ for $f_0 = 2$, $\alpha = 0.2$, $\delta = 1$ and $\tau = 1$ and put them together to find the solution plotted in Fig. 15.10a.

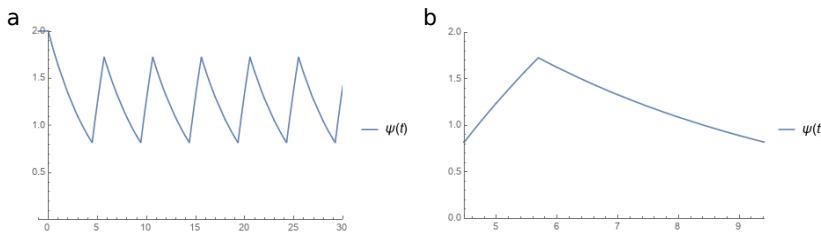


Fig. 15.10

UNSAT-FFF delay ODE model. (a) Solution of (15.23) using the recursion (15.30) on $t \in [-\tau, 30\tau]$ for $f_0 = 2$, $\alpha = 0.2$, $\delta = 1$ and $\tau = 1$. (b) One period of the oscillation of solution in (a) consisting of two exponential pieces. Figure reproduced from (Leifer et al., 2020). Copyright © 2020, Leifer et al.

By (15.30), all functions f_k are the sum of an exponential function and a constant. By (15.23), when the Heaviside function is equal to zero we get a solution that decays exponentially to zero. Likewise, when the Heaviside function is equal to 1 we get a solution that grows exponentially to $\frac{\delta}{\alpha}$. In other words, the solution grows until $\theta(1 - \psi(t - \tau))$ changes to zero (i.e., when $\psi(t - \tau) > 1$), and decays until $\theta(1 - \psi(t - \tau))$ changes to 1 (i.e., when $\psi(t - \tau)$ crosses 1 again, but from the other side). Therefore we get oscillations consisting of two exponential pieces. One period of the oscillation is shown in Fig. 15.10b. The solution on this interval is given by:

$$\begin{aligned} \psi(t) &= 5 - 10.2 * e^{-0.2t} \text{ for } t \in [4.47, 5.69] \\ \psi(t) &= 5.4 * e^{-0.2t} \text{ for } t \in [5.69, 9.42], \end{aligned} \quad (15.31)$$

which is the predicted behavior. This circuit functions like a capacitor, charging and discharging in an RC circuit.

15.3.5 UNSAT-FFF clock functionality

As shown above, the UNSAT-FFF has the functionality of a clock, a fundamental unit in any computing system. The solution of the discrete-time Boolean interaction model for $\lambda > 1$ oscillates in time, and so does the DDE considered in the previous section.

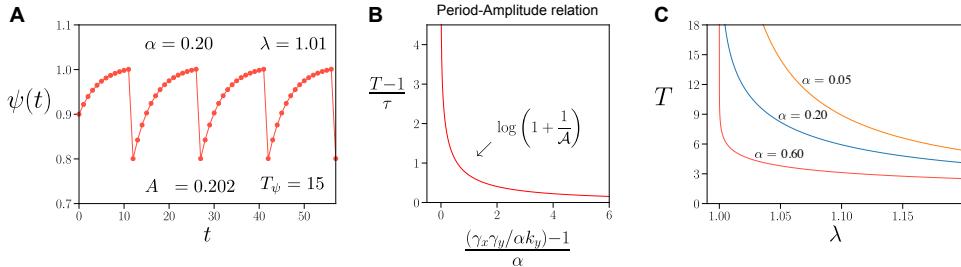


Fig. 15.11

Robust UNSAT-FFF clock functionality. (a) Solutions for $\alpha = 0.2$ and $\lambda = 1.01$. The values for $A_\psi = 0.202$ and $T = 15$ obtained from (15.33) and (15.35) agree perfectly with those found by numerical simulation. (b) Period-amplitude relationship in terms of the original set of parameters α , k_y , γ_x , and γ_y . (c) Period of oscillations as a function of λ for different values of α . Figure reproduced from (Leifer et al., 2020). Copyright © 2020, Leifer et al.

To compute the amplitude of oscillations A_ψ for the rescaled variable ψ_t , we recall that the iterative map $\psi = f(\psi)$ satisfies the recursive equation:

$$f^t(\psi) = f^{t-1}(\beta\psi)\theta(\psi - 1) + f^{t-1}(\beta\psi + \alpha\lambda)\theta(1 - \psi). \quad (15.32)$$

The amplitude of oscillations A_ψ is given by

$$A_\psi = \lim_{\psi \rightarrow 1^-} f(\psi) - \lim_{\psi \rightarrow 1^+} f(\psi) = \alpha\lambda, \quad (15.33)$$

which implies that

$$A_\psi = \frac{\gamma_x\gamma_y}{k_y}. \quad (15.34)$$

To find the period T of the oscillations, we recall from (15.21) that the solution for the minimum value of ψ , $\psi_{\min} < 1$, evolves to its maximum value ψ_{\max} in $T - 1$ iterations as $\psi_{\max} = e^{-(T-1)/\tau}\psi_{\min} + \lambda(1 - e^{-(T-1)/\tau})$. Since $\psi_{\min} = (1 - \alpha)\psi_{\max}$, because $\psi_{\max} > 1$, we find:

$$T = \left\lceil 1 + \tau \log \left(1 + \frac{\alpha}{\lambda - 1} \right) \right\rceil, \quad (15.35)$$

where we used $\psi_{\max} = 1$. For example, using $\alpha = 0.2$ and $\lambda = 1.01$, we find $A_\psi = 2.02$ and $T = 15$, which agrees with the numerical simulation in Fig. 15.11a.

Equation (15.35) lets us define a rescaled amplitude $\mathcal{A} = (\lambda - 1)/\alpha$ and a reduced period $\mathcal{T} = (T - 1)/\tau$ such that

$$\mathcal{T} = \log \left(1 + \frac{1}{\mathcal{A}} \right), \quad (15.36)$$

which corresponds to the *period-amplitude relationship* of the UNSAT-FFF. A plot of this relationship is shown in Fig. 15.11b, where we plot $(T - 1)/\tau$ as a function of $[(\gamma_x\gamma_y/\alpha k_y) - 1]/\alpha$.

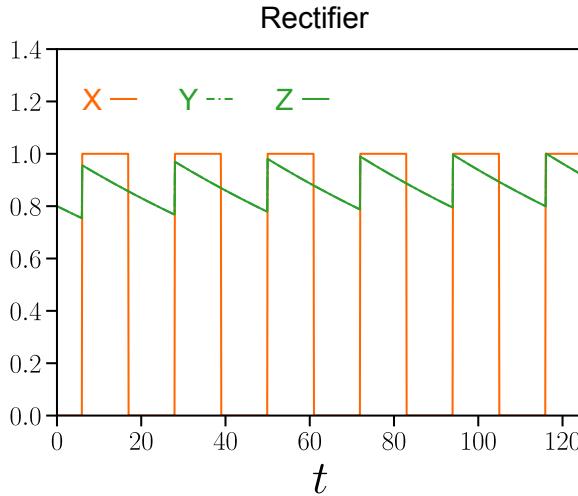


Fig. 15.12

UNSAT-FFF clock functionality. The intrinsic relationship between the period T and amplitude of oscillations of the UNSAT-FFF provides a stabilizing effect which results in predictable and robust oscillations. Here, the UNSAT-FFF operates as a rectifier for the set of parameters: $\alpha = 0.001$, $k_x = 0.5$, $k_y = 0.8$, $\gamma_x = 0.45$, and $\gamma_y = 0.45$.

Coming back to the original variable $y_t = k_y \psi_t$, the amplitude of oscillation of y_t , namely $A = k_y A_\psi$, is given by:

$$A = \gamma_x \gamma_y, \quad (15.37)$$

From (15.35) we can write the period of oscillation as a function of the original parameters:

$$T = 1 - \frac{1}{\log(1 - \alpha)} \log \left(1 + \frac{\alpha}{(\gamma_x \gamma_y / \alpha k_y) - 1} \right). \quad (15.38)$$

Figure 15.11c shows the period of oscillation T as a function λ for $\alpha = 0.60$, $\alpha = 0.20$, and $\alpha = 0.05$.

The clock functionality of the UNSAT-FFF can be understood by analyzing its response function, i.e. the relation between oscillations at the input and at the output of the circuit. The amplitude A_y , and period T of the oscillations are not independent, as in a harmonic oscillator, but are related through the ‘period-amplitude’ relation expressed by (15.36) and Fig. 15.11b. From (15.38), for α sufficiently small,

$$T - 1 \sim \frac{k_y}{\gamma_x \gamma_y} = \frac{1}{A}, \quad (15.39)$$

which constrains the ‘clock’ (T) of the circuit to the power (A_y). As a consequence, A_y and T cannot be controlled arbitrarily, and this (A-T) constraint helps to stabilize the UNSAT-FFF response against disturbance in the input X .

For example, for a given available power supply, the system is constrained to dissipate this power, and when the UNSAT-FFF oscillates, it is automatically set to operate on an

extended time window (T large) at low amplitude A when a small expression level is required (A small) and vice-versa.

If the signal from X has a frequency f_x much higher than the oscillator, the modulator Y will transmit a signal oscillating with very small amplitude $A \propto 1/T_x$. In other words, the UNSAT-FFF can also operate like a rectifier for high-frequency signals coming from X. For example, Fig. 15.12 shows the UNSAT-FFF operating as a rectifier for the set of parameters: $\alpha = 0.001$, $k_x = 0.5$, $k_y = 0.8$, $\gamma_x = 0.45$, and $\gamma_y = 0.45$. Similarly, if the frequency f_x of the baseband signal x_t is small compared to f_y , the modulator will transmit a signal with an amplified bandwidth of order f_y .

The stabilizing effect of the negative feedback loop makes synchronization predictable and robust through a symmetry fibration and endows the UNSAT-FFF circuit with a reliable functionality independently of fine-tuning of parameters. This is ideal for bacterial TRNs, which need to adapt to large variation in environmental and growth conditions. This reliability, together with the simplicity of its feedback structure, may explain why the FFF is ubiquitous in simpler genomes across the bacterial domain like *B. subtilis*, *E. coli*, *Salmonella* and *M. tuberculosis*, although they are not present in eukaryotes (Morone et al., 2020).

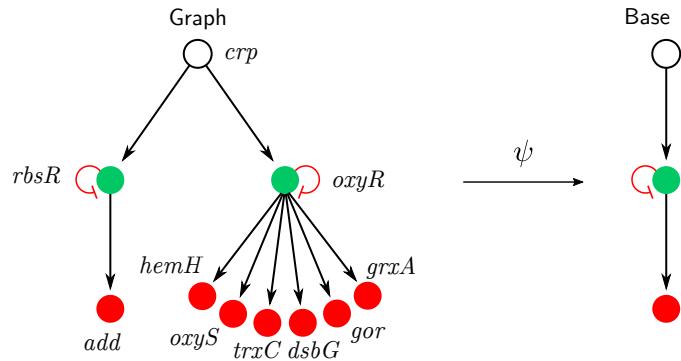
The findings related to clock functionality can be applied to other oscillatory building blocks, such as the Fibonacci fiber and the $n = 2$ fiber, which we will explore next. The concept is that Fibonacci fibers with increasingly longer cycles can support more robust oscillatory patterns compared to the basic autoregulation negative feedback loop found in the FFF.

15.4 Simple multilayer fiber

Synchronization of the genes within the FFF building block is guaranteed by their isomorphic input trees. These input trees contain infinitely many layers. However, for this particular circuit synchronization is guaranteed by the first layer of the input tree, the input set, because all genes in the fiber have the same in-neighbors. Thus, while the input tree is infinite, only the isomorphism of the first layer is necessary to ensure synchronization. The next level of complexity in the hierarchy of fibers is circuits, where synchronization depends on deeper input layers of synchronized genes and longer loops of information. This increases the complexity of the circuits since the synchronization between genes connects coherently distant areas in the network. We call this building block the *Simple Multilayer Fiber*.

Another way to increase the complexity is by the appearance of longer cycles of information, as in the Simple Fibonacci circuits explained in the next section. We call these circuits ‘simple’ since the multilayers are composed of genes in single fibers. Complex multilayer fibers made of multiple fibers are treated in Chapter 16. Multilayer and Fibonacci fibers are important in bacteria, although they appear in low numbers. They are very abundant in more complex species, such as eukaryotes, from yeast to humans. This compares to the

a $|n = 0, l = 1\rangle \oplus |n = 1, l = 1\rangle$: Multi-layer composite fiber



b Input trees

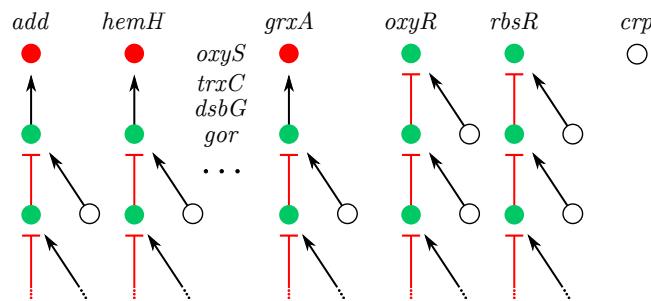


Fig. 15.13

Simple Multilayer Fiber. (a) Circuit consists of two layers of fibers: *add*, *dsbG*, *gor*, *grxA*, *hemH*, *oxyS*, *trxC* classified as $|n = 0, l = 1\rangle$ and *rbsR*, *oxyR* classified as $|n = 1, l = 1\rangle$, therefore forming a multi-layer composite fiber $|n = 0, l = 1\rangle \oplus |n = 1, l = 1\rangle$. The fibration ψ of this circuit collapses both fibers and leaves the regulator untouched. (b) Genes in the red fiber receive one input from the gene in the green fiber, which in turn receives an input from itself and the regulator. Therefore, input trees of genes in the red fiber resemble the sum of an input tree of $|n = 0, l = 1\rangle$, followed by the input tree of $|n = 1, l = 1\rangle$. Input trees of the green fiber are those of the FFF. The regulator node has no inputs. Thus, the multi-layer composite has two nontrivial fibers that can synchronize their activity. The gene *add* is separated from the rest of the red fiber by two steps, therefore allowing for long-range synchronization in the network. Figure reproduced from (Leifer et al., 2021). Copyright © 2021, The Author(s).

simpler AR and FFF (and operons), which are more prevalent in bacteria (Leifer et al., 2020).

Figure 15.13a shows an example of a multilayer composite fiber in *E. coli* whose main regulator is *crp*. In this case, *crp* is the inducer of a composite fiber, composed of *oxyR*

and *rbsR* and responsible for further downstream regulation of several carbon utilization subsystems of genes.

The input tree of this fiber is isomorphic to that of FFF $|1, 1\rangle$, even though the building block has a very different topology from that of the FFF, Fig. 15.4a. This first layer of genes regulates, via *oxyR* and *rbsR*, a second fiber composed of genes *add*, *dsbG*, *gor*, *grxA*, *hemH*, *oxyS*, *trxC*. If the branch corresponding to *rbsR* is disregarded, the building block of the fiber of genes *dsbG*, *gor*, *grxA*, *hemH*, *oxyS*, *trxC* is classified as a single layer $|0, 1\rangle$. Thus the building block corresponding to the entire fiber in Fig. 15.13a in red is a double-layer composite, which we denote by $|add - oxyS\rangle = |0, 1\rangle + |1, 1\rangle$, made of a series of genes composing a single fiber of type $|0, 1\rangle = |add, dsbG, gor, grxA, hemH, oxyS, trxC\rangle$, which is regulated by two different transcription factors *rbsR* and *oxyR* that form another fiber of type $|1, 1\rangle = |rbsR, oxyR\rangle$. This composite is important because it allows information to be shared between two genes, *add* and *oxyS*, which are not directly connected.

The gene *add* is two edges away from the other genes in its fiber, thus achieving synchronization at a distance of two in the network. This is a clear example of global communication and integration of unconnected genes in the network structure. The input trees and base of this multilayer composite, depicted in Fig. 15.13a, show that this composite fiber is the union of two fundamental fibers.

Composite fibers satisfy a simple engineering ‘sum-rule’: elementary fibers are composed as a series of fibers in a predefined order where the first layer is represented by an entry fiber (carrying transcription factors), and the last layer is formed by a terminator fiber of output genes (encoding enzymes). This multi-layer composite fiber is biologically significant because genes in the output layer synchronize a genetic module that implements the same function even though the genes in the module can be far apart in the network. Such functionally related modules could not be identified by typical modularity algorithms, which cluster nodes in modules of highly connected nodes.

We find that composite fibers are dominant in eukaryotes (yeast, mouse, human). They resemble the building blocks of multilayered deep neural networks where each subsequent gene in the layer synchronizes, even though nodes can be distant in the network. More generally, composite fibers with multiple layers streamline the construction of larger aggregates of fibration building blocks, performing more complex functions in a coordinated fashion.

The biological relevance of these fibers results from the existence of genes that can synchronize even though they do not share any common inputs. So, what role does this composite fiber play in the regulatory network? Why has nature not chosen to give them the same regulatory inputs in a single layer? The main functionality of the multilayer fibers is to communicate via synchronization of different fibers in the network that are separated by long distances. Multilayer fibers are the predominant method to achieve distant synchronization, indicating the higher level of complexity in these composite circuits. This structure is critical to integrate information in the large-scale network.

AR loops and FFF are commonly found in bacteria, while we observe only one type of multilayer fiber, as shown in Table 14.2. Multilayer fibers are more prevalent in higher organisms, such as yeast and humans. This is supported by the analysis of fibers across various species in (Morone et al., 2020; Leifer et al., 2020), which highlights the increased complexity in the network structure and functionality of eukaryotes.

15.5 Simple Fibonacci fibers

The next stage in the hierarchy is the Simple Fibonacci fiber (FF) of Fig. 15.14a (Morone et al., 2020; Leifer et al., 2020). The FF shows a higher level of complexity in the paths that regulate the fiber. To understand the FF, and also the more complex FF that we discuss in Chapter 16, we invoke the strongly connected component of Definition 3.8. In general, a fiber may receive information from the entire network through its input tree. When the fiber is not within an SCC, information is processed only inside the fiber. This is the case for all fibers described so far and characterized by integer fiber numbers $n = 0, 1$. The fiber may still receive signals from the SCC, such as the case of the studied fibers so far. But, the fiber does not send back signals, so the fiber, or any gene in the fiber, is not part of the SCC.

However, if a fiber is connected to an SCC and sends back information to its own regulators through the SCC, the level of complexity of the fiber topology increases. In previous examples, the input trees are infinite due to AR loops; here, the input tree becomes infinite due to longer cycles in the SCC of the network. That is information cycles around a longer loop before returning to the fiber. The input tree of the fiber contains longer cycles of information arriving at the root gene. These cycles introduce extra terms in the sequence layers in the input tree, leading to Fibonacci sequences in the number of paths. There is an infinite number of possibilities for these cycles to appear in a network. The simplest one is the Simple Fibonacci fiber.

From the starting point of a $|1, 1\rangle$ FFF, many different circuits can correspond to this base $|1, 1\rangle$. However, only certain modifications conserve the topological class identified by $|1, 1\rangle$. For instance, changing the sign of the edges is allowed as long as the edges inputting to each gene are the same, but removing the edge from the regulator to one of the genes in the fiber will break the symmetry of the fiber, so it is not allowed.

Adding a second gene downstream of the AR in the FFF will conserve the topological class, but only if it interacts with X following lifting. This situation changes as soon as the fiber feeds information back to the external world. This is the case for the Fibonacci fiber in Fig. 15.14, where the gene *uxuR* now regulates its regulator gene *exurR*. The inclusion of this second feedback loop, the shortest of such a feedback loop from gene *uxuR* to its regulator, forms a second cycle of length $d = 2$, resulting in the coexistence of two timescales in the network. This, in turn, increases the diversity of trajectories and delays in the network, a dynamic complexity that is measured by the divergence of the branching ratio of the input tree, which is captured by the sequence a_t .

The building block of the fiber *uxuR-lgoR* that is regulated by the connected component *crp-fis* (Fig. 15.14a) forms an intricate input tree (Fig. 15.14b) where the number of paths of length $i - 1$ is encoded in a Fibonacci-like sequence, the *Lucas numbers*:

$$a_i = 1, 3, 4, 7, 11, 18, 29, \dots \quad (15.40)$$

This sequence leads to a non-integer branching ratio, the golden number φ :

$$\lim_{t \rightarrow \infty} \left(\frac{a_{t+1}}{a_t} \right) = \varphi = \frac{1 + \sqrt{5}}{2} = 1.6180\dots \quad (15.41)$$

a $|\varphi_d = 1.6180..., \ell = 2\rangle$: Fibonacci fiber

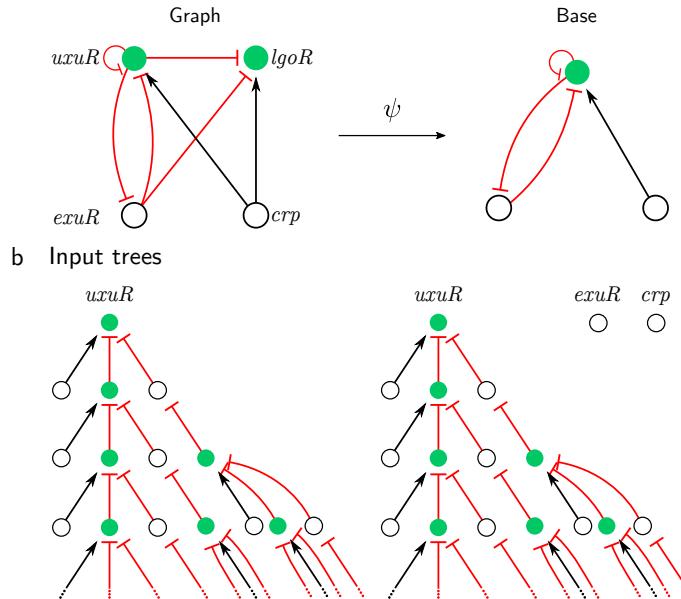


Fig. 15.14

Simple Fibonacci fiber (FF). (a) (left) FF circuit is the FFF circuit with the additional edge from the fiber back to the regulator. In this example *uxuR* sends back to *exuR*, creating an extra loop in the circuit. The extra edge does not change the fiber. (a) (right) FF circuit is the FFF circuit with the fiber, so the fibration stays the same. (b) However, the extra loop changes the input tree of fiber nodes. *uxuR* receives from itself and *exuR*, which in turn receives from *uxuR*, which creates an input tree with layer sizes following the Fibonacci sequence. The branching ratio then defines the first fiber number, so this FF is classified as $|\varphi, \ell = 2\rangle$, where $\varphi = 1.6180$. Node *lgoR* receives an input from *exuR* and then from *uxuR*, which means that even if there were no link from *uxuR* to *lgoR*, information would still be passed along through the regulator. This is another way for networks to process information. Figure reproduced from (Leifer et al., 2021). Copyright © 2021, The Author(s).

revealing that the input tree follows a Fibonacci recursion

$$a_t = a_{t-1} + a_{t-2}, \quad (15.42)$$

updating the current state two steps backwards.

The repressor interactions between genes $X = uxuR$, $Y = exuR$, and $Z = lgoR$ function exactly as a Fibonacci fiber (Fig. 15.14a). The important component of these circuits is the delay in the feedback loop through the regulator from $Y \rightarrow X$ and back to Y captured by the a_{t-2} term in the Fibonacci sequence.

This topology arises in the genetic network due to the combination of two cycles of

information flow. First, the autoregulation loop inside the fiber at *uxuR* creates a cycle of length $d = 1$, which contributes to the input tree an infinite chain with branching ratio $n = 1$. This sequence is reflected in the Fibonacci series by the term $a_i = a_{i-1}$. The important addition to the building block is a second cycle of length $d = 2$ between *uxuR* in the fiber and its regulator *exuR*: $\text{uxuR} \rightarrow \text{exuR} \rightarrow \text{uxuR}$. This cycle sends information from the fiber to the regulator and back to the fiber by traversing a path of length $d = 2$ that creates a delay of $d = 2$ steps in the information that arrives back at the fiber (Fig. 15.14b). This short-term memory effect is captured by the second term $a_i = a_{i-2}$ in the Fibonacci recursion, leading to $a_i = a_{i-1} + a_{i-2}$ and the golden ratio limit.

Morone et al. (2020) found three types of Fibonacci fibers in the TRN of *E. coli*, as seen in Fig. 14.10. Eukaryotes like yeast and humans present a much richer variety of Fibonacci fibers (Leifer et al., 2020). These circuits have been synthetically implemented by Stricker et al. (2008) using a hybrid promoter that drives the transcription of genes *araC* and *lacI* forming a dual-feedback circuit. The functionality of this circuit has been demonstrated to be robust oscillations, due to the negative feedback loop (Stricker et al., 2008). We show in Chapter 18 that symmetry breaking in this Fibonacci circuit leads to the base of the JK flip-flop, which is the universal storage device of computer memories.

The above argument implies that an autoregulated fiber that further regulates itself by connecting to its connected component via a cycle of length d encodes a generalized Fibonacci sequence of order d defined as:

$$a_i = a_{i-1} + a_{i-d}, \quad \text{generalized Fibonacci sequence} \quad (15.43)$$

with generalized golden ratio φ_d (Fig. 14.10, fourth row).

We find such a Fibonacci sequence in the *evgA-nhaR* fiber building block (Fig. 14.10, third row) linked to the pH strongly connected components shown in Fig. 14.4. This fiber contains an autoregulation cycle inside the fiber and also an external cycle of length $d = 4$ through the pH strongly connected component: $\text{evgA} \rightarrow \text{gadE} \rightarrow \text{gadX} \rightarrow \text{hns} \rightarrow \text{evgA}$ (Fig. 14.10b, third row).

This topology forms a fractal input tree with sequence:

$$a_i = a_{i-1} + a_{i-4} \quad \text{generalized Fibonacci sequence of fiber evgA} \quad (15.44)$$

and branching golden ratio of a generalized Fibonacci sequence:

$$\varphi_4 = 1.38028\dots \quad (15.45)$$

Interestingly, this sequence has been indexed by the *The On-Line Encyclopedia of Integer Sequences* as sequence A123456 in (Sloane, 2020). We call this topology the φ_4 -Fibonacci fiber.

Generalized Fibonaccis also appear inside strongly connected components, like the *rcsB-adiY* φ_3 -FF in the pH system (Fig. 14.10, second row).

As a final generalization, if the fiber participates in many hierarchical cycles with the regulators of varying lengths up to a maximum d , the Fibonacci sequence generalizes to:

$$a_i = a_{i-1} + a_{i-2} + \dots + a_{i-1-d} + a_{i-d}, \quad (15.46)$$

and the branching ratio φ_d satisfies

$$d = -\frac{\log(2 - \varphi_d)}{\log \varphi_d}. \quad (15.47)$$

In a Fibonacci fiber, genes connect to the SCC, and the SCC connects back to the fiber via directed paths. If the SCC contains many paths then, in principle, the building block associated with the FF should contain all these paths through the SCC. To define a building block associated with a Fibonacci fiber, we consider only the shortest cycle from the fiber through the SCC leading back to the fiber. Longer paths through the SCC are not included in the building block. Consideration of these neglected cycles in building blocks could substantially affect the dynamics of gene expression in the fiber. However, they do not affect the synchronization of genes in the FF.

15.6 Binary tree fiber ($n = 2$) and n -ary tree fibers

The last type of simple building block in the hierarchy of fibers that we find in the TRN of *E. coli* is the *binary tree fiber* with $n = 2$, shown in Fig. 15.2e with a core $|2, l = 0\rangle$. It is characterized by two AR loops, leading to a symmetric input tree; such a circuit is shown in Fig. 15.2e. This figure shows an $n = 2$ binary tree fiber resulting from the repressor regulations between genes $X = lexA$ and $Y = rocR$, and from the positive regulations between genes $X = hprT$, $Y = tilS$, and $Z = ftsH$, both from the *B. subtilis* TRN studied in (Álvarez-García et al., 2025a).

The sequence of information is coded in a sequence defined by

$$a_i = 2a_{i-1}, \quad \text{binary tree fiber sequence} \quad (15.48)$$

and we classify this fiber as $n = 2$. This procedure can be iterated to any number of loops forming n -ary trees, as shown in the last row of Fig. 14.9, but the bacterial networks we have studied do not contain any fiber with $n > 2$, suggesting a practical limit in complexity for these organisms.

15.7 Evolutionary dynamics of fibration building blocks

We have provided a constructive mechanism that reveals the hierarchy of symmetric building blocks. Their broken symmetry counterparts, which are also important, are studied in Chapter 18. The procedure mimics an evolutionary growth procedure by recursively iterating the process that expanded from a primordial AR loop to more complex circuits by extensions to longer feedback loops producing functionalities like synchronized clocks and oscillators. In parallel, a process of symmetry breaking of lifted circuits creates further

functionalities beyond synchronization and oscillations, providing logic computations and memory storage through broken symmetry states, see Chapter 18.

Summarizing, the starting point of an evolutionary process of growing circuits is the primordial AR loop with no external regulator and no regulated genes:

$$|1, l = 0, m = 0\rangle, \quad \text{primordial AR base.} \quad (15.49)$$

This circuit can grow by lifting to m regulated genes $|1, l = 0, m = 0\rangle$ following a transition:

$$|1, l = 0, m = 0\rangle \rightarrow |1, l = 0, m \neq 0\rangle \quad \text{lifting of the AR base.} \quad (15.50)$$

Next, the bare AR base, (15.49), can receive a regulator from another gene forming the base of a new circuit, the FFF, following a transition:

$$|1, l = 0, m = 0\rangle \rightarrow |1, l = 1, m = 0\rangle \quad \text{from AR base to FFF base.} \quad (15.51)$$

This does not yet affect the complexity because all the relevant dynamics remain constrained to the sole loop in the FFF circuit. The next transition is again by lifting this FFF base to produce the FFF with m regulated genes:

$$|1, l = 1, m = 0\rangle \rightarrow |1, l = 1, m \neq 0\rangle \quad \text{lifting of the FFF base.} \quad (15.52)$$

These modifications conserve the topological class of the FFF, providing its synchronization. Evolutionarily, the lifting property can be thought of as a duplication mechanism (as discussed in Section 18.3) that conserves dynamic synchronization. A duplication process produces more coherent genes that might be able to reproduce a functionality or could be free to mutate and diverge into other functions.

A transition to a new class of building blocks occurs as soon as the autoregulated gene in the FFF feeds information back to the external regulator, converting the FFF base into an FF base:

$$|1, l = 1, m = 0\rangle \rightarrow |\varphi_2 = 1.6180, l = 1, m = 0\rangle \quad \text{from FFF base to FF base.}$$

This second feedback loop results in the coexistence of two timescales in the network: one of the AR loop and another of length two, resulting in the simple Fibonacci sequence of the input tree. A new transition to the full FF is produced by lifting again:

$$|\varphi_2 = 1.6180, l = 1, m = 0\rangle \rightarrow |\varphi_2 = 1.6180, l = 1, m \neq 0\rangle \quad \text{lifting of the FF base.}$$

The AR loop and an additional short cycle to the regulator still dominates the dynamics of this FF. This cycle can be enlarged with the supposed improvement in functionality: oscillators like the FF and FFF are known to be more stable when the feedback loops are longer. Besides, longer cycles to the regulator produce longer memory in the dynamics. Thus the next steps include longer and longer cycles to the regulator. These building blocks are generalized FFs with path length or regulatory cycles d . The first transition is from $d = 2$, the simple FF φ_2 -FF to the generalized φ_3 -FF of length $d = 4$, as seen in *E. coli* Fig. 14.10, second row:

$$|\varphi_2 = 1.6180, l = 1, m = 0\rangle \rightarrow |\varphi_3 = 1.4655, l = 1, m = 0\rangle \quad \text{from } \varphi_2\text{-FF to } \varphi_3\text{-FF base.}$$

We consider the bases which can then be lifted to produce a variety of regulated genes. The next natural transition observed in *E. coli* is to the φ_4 -FF depicted in Fig. 14.10, third row:

$$|\varphi_3 = 1.4655, l = 1, m = 0\rangle \rightarrow |\varphi_4 = 1.3802, l = 1, m = 0\rangle \quad \text{from } \varphi_3\text{-FF to } \varphi_4\text{-FF base.}$$

While the evolution of the FF in bacteria ends here, we can imagine longer and longer cycles generated by adding one intermediate gene at a time to produce longer and longer memory effects.

$$|\varphi_d, l = 1, m = 0\rangle \rightarrow |\varphi_{d+1}, l = 1, m = 0\rangle \quad \text{from } \varphi_d\text{-FF to } \varphi_{d+1}\text{-FF base.}$$

When $d \rightarrow \infty$ we find that $\varphi_\infty \rightarrow 1$. The feedback loop effectively disappears and the resulting φ_∞ -FF turns into a FFF with $n = 1$.

The final transition in *E. coli* occurs when the regulator in the FF acquires an AR loop, effectively turning it into a binary-tree fiber with $n = 2$:

$$|\varphi_2, l = 1, m = 0\rangle \rightarrow |n = 2, l = 1, m = 0\rangle \quad \text{from } \varphi_2\text{-FF to binary-tree base,}$$

which can then be lifted to create more duplicated genes regulated by the binary-tree base:

$$|n = 2, l = 1, m = 0\rangle \rightarrow |n = 2, l = 1, m \neq 0\rangle \quad \text{lifting of the binary-tree base.}$$

While this systematic process can generate ever more complicated circuits, the bacterial genome evolution stops here. We quantify their complexity next.

15.7.1 Complexity of the building blocks: fractal dimensions

The complexity of the Fibonacci fiber with feedback to the regulator is the branching ratio 1.6180..., which tends back to 1 when $d \rightarrow \infty$. This golden ratio can be thought of as the *fractal dimension* of the input tree. The simplest building blocks have integer dimensions $D = 0$ (a regulon studied below in Section 15.8), $D = 1$: the AR loop fiber and the FFF and $D = 2$: the binary-tree fiber. The complexity of the AR loop and the FFF is the same; their only difference is the external regulator. The complexity class changes dramatically in the FF class, which is now described by their fractal dimensions φ_d , where $d = 2$ for the simple FF generalized to any d and tending to the FFF as $d \rightarrow \infty$. From $d = 2$ to $n = 2$, the next level from FF to a binary tree, there is a discontinuity in fractal dimension values.

The fractal dimension of the FF is lower than the number of loops in the circuit (two). The intuition of what this reveals is that, in this circuit, the regulator is still not part of the base since it does not receive input from itself. Hence, it is not within the fibration symmetry of the fiber. This, in turn, indicates naturally that the next element in the hierarchy of fibers results from the inclusion of an AR loop leading to the binary-tree fiber, with complexity exactly 2.

The increasing complexity observed has an evolutionary basis rooted in the mechanism of gene duplication by lifting, which creates more connections, or edges, between existing genes. The fractal dimensions identified correspond to different classes of circuits, similar to how dimensions define universal classes of matter in physics. For instance, in the case of the simplest bacterial genome associated with the TRN, the fractal dimensions stop at 2. In Chapter 16, we explore the symmetries of metabolism, which is characterized

by an extensive number of cycles. These cycles are organized in a manner that leads to progressively higher fractal dimensions, reaching an impressive maximum of 25.97, which indicates increasingly greater complexity.

15.8 Biologically trivial fibers: operons and regulons

The trivial building blocks leading to synchronization are operons and regulons. We recall these concepts. Operons are ubiquitous in bacteria (Jacob and Monod, 1961): genes in an operon have a common promoter and are not transcribed into individual mRNAs but as transcription units, yielding a single mRNA strand containing several contiguous genes. Depending on the locations of promoters and terminators, the transcription units can also overlap. The expression of genes in an operon is automatically synchronized since they are translated together. In the case of multi-promoter operons, we can group the genes into minimal transcription units, each being controlled by the same combination of promoters. The genes in such a minimal unit should be precisely coexpressed (operons with several terminators can be subdivided similarly). In our analysis, we take the operon as a single node in the TRN. When two or more TFs belong to the operon, we leave one TF associated with the operon and separate the remaining TFs from the operon. While the synchronization in the operon is important biologically, from the point of view of fibrations, they are trivial, since their synchronization comes from a single promoter region and polymerase.

Beyond operons, the next trivial network structure that can synchronize genes is the regulon, defined as the set of genes regulated by a single TF. This kind of circuit is abundant in the *E. coli* TRN, see Table 14.2. Figure 15.15a shows an example, the regulon of the transcription factor Fis in *E. coli*. A note on notation: In bacteria, *fis* refers to the gene while Fis denotes the protein. Typically, there is one protein per gene, but eukaryotes exhibit variation due to splicing.

The regulon contains four units: the operon *cbpAM* and the genes *gltX*, *gyrB*, *msrA*. Gene *fis* is, in turn, regulated by Crp. Crp and Fis are the two master regulators of carbon utilization. The regulon is an example of the most trivial form of symmetry. Assuming that the genes have no other regulators, the symmetry can be captured by an automorphism of a simple permutation in the regulon, for instance *cbpAM* \leftrightarrow *gltX*, or any permutation of the four genes in Fig. 15.15a.

These trivial fibration circuits are the same as some network motifs. In the motif nomenclature of (Alon, 2019) the $|0, 2\rangle$ fiber shown in Fig. 15.15b is a *FAN motif*, while the $|0, 1\rangle$ fiber depicted in Fig. 15.15a is a *star motif*.

The symmetry group of this building block formed by the two symmetric genes is the symmetric group \mathbb{S}_4 (recall Definition 4.4). Here, the symmetry group \mathbb{S}_4 of the regulon in Fig. 15.15a contains all $4! = 24$ permutations of the four genes. The genes in the regulon are thus synchronized in orbits generated by automorphisms, so the regulon is also a fiber. The symmetry group \mathbb{S}_4 confirms the known fact that the genes in the regulon, namely *cbpAM*, *gltX*, *gyrB*, and *msrA*, synchronize. However, these genes do not synchronize with

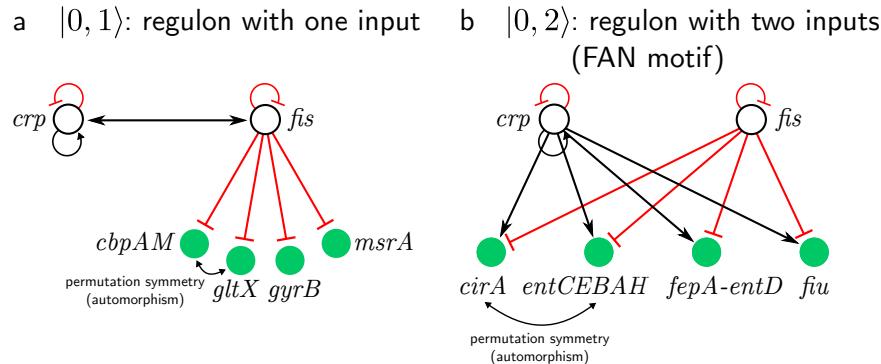


Fig. 15.15

Trivial fibration building blocks: Regulons with co-regulation. (a) Genes *cbpAM*, *gltX*, *gyrB* and *msrA* are controlled by the same TF (*fis*). Fiber numbers describing this circuit are $|n = 0, l = 1\rangle$ since there are no loops and fiber has one regulator. Gene activity can synchronize because any two nodes can be permuted without any change in the network under the \mathbb{S}_4 symmetry group. (b) Regulon circuit consisting of genes *clrA*, *fiu* and operons *entCEBAH*, *fepA-entD* controlled by two regulators *crp* and *fis* also synchronizes, because the symmetry group S_4 is preserved irrespective of the number of regulators. The fiber has two regulators and no cycles, and therefore is characterized by fiber numbers $|n = 0, l = 2\rangle$. Figure reproduced from (Leifer et al., 2021). Copyright © 2021, The Author(s).

the regulator *fis* since the regulator does not belong to the orbit. This is because it receives an edge from *crp*, thus breaking the fibration symmetry with its regulon.

When the genes in a regulon are also under the control of other TFs, there is a chance that the regulons might also preserve the automorphism symmetries. For instance, the single-regulon circuit controlled by *fis* (Fig. 15.15a) can be augmented by a second regulator as in Fig. 15.15b. The same symmetric group \mathbb{S}_4 describes the symmetry among the genes *clrA*, *fiu* and operons *entCEBAH*, *fepA-entD* since all of them can be permuted. These genes are synchronous but not with the regulators *crp*, *fis*.

These circuits are characterized by fiber numbers $|n = 0, l\rangle$ since they have no loops inside the fiber ($n = 0$), and l external regulators. These circuits do not need the input tree to be characterized. This is because, in a circuit without cycles, $n = 0$, only the regulator is needed to characterize the dynamical state of synchronization, so the input tree becomes the input set, revealing their triviality.

15.9 Fibration building block landscape across networks and species

In previous sections, we have shown that fibration building blocks are organized in a hierarchy that can be classified using fiber numbers, the branching ratio with integer and fractal dimensions, and the number of regulators and regulated genes. This hierarchy is mainly based on bacterial genetic networks. Morone et al. (2020) extended these results to complex networks of different domains and biological networks of different species. The fiber finding algorithm of Section 13.7 was applied to 373 datasets across different domains. Among biological networks, the analysis has been performed on neural networks, gene regulatory networks, disease networks, signaling pathways, and metabolic networks. Networks of different species have been studied, including the bacteria *E. coli*, *B. subtilis*, *Salmonella enterica* and *Mycobacterium tuberculosis*, along with networks of *Arabidopsis thaliana* and *Drosophila melanogaster* and of higher complexity like yeast *Saccharomyces cerevisiae*, mouse *Mus musculus*, and human *Homo sapiens* networks.

Figure 15.16a and b show that fibration building blocks are present in all the species mentioned above and types of networks. Distributions of the building block fiber numbers (including multi-layer and Fibonacci) across types of biological networks, summed over species, are shown in Fig. 15.16a. Figure 15.16b shows distributions of the building block fiber numbers across different species summed over the type of the network. A common feature of these networks is that the number of fibers decreases as l increases, for both $n = 0$ and $n = 1$. Another interesting observation is that bacteria contain a higher proportion of $n = 1$ building blocks compared to more complex organisms like yeast, mice, and humans, which have a higher portion of Fibonacci and multi-layered composite building blocks. One possible explanation might be that nodes in networks of bacteria have more self-loops, while networks of mice and humans contain longer cycles in their networks, creating more elaborated building blocks.

Beyond biological networks, Morone et al. (2020) examined networks in various domains, including ecosystems, the internet, software, economics, social interactions, and infrastructure. The distributions of fiber numbers for the building blocks studied are presented in Fig. 15.16c. Additionally, the normalized sum of all distributions across these domains and species is illustrated in Fig. 15.16d, which shows the cumulative number of fibers for all domains and species per network size of 10,000 nodes. The results indicate the presence of symmetry fibrations, supporting the existence of fibration building blocks in these networks across multiple domains and species. This research serves as a foundational step toward applying the concept of symmetry fibration beyond biological contexts, helping to describe the building blocks of networks in any domain where synchronization plays a significant role.

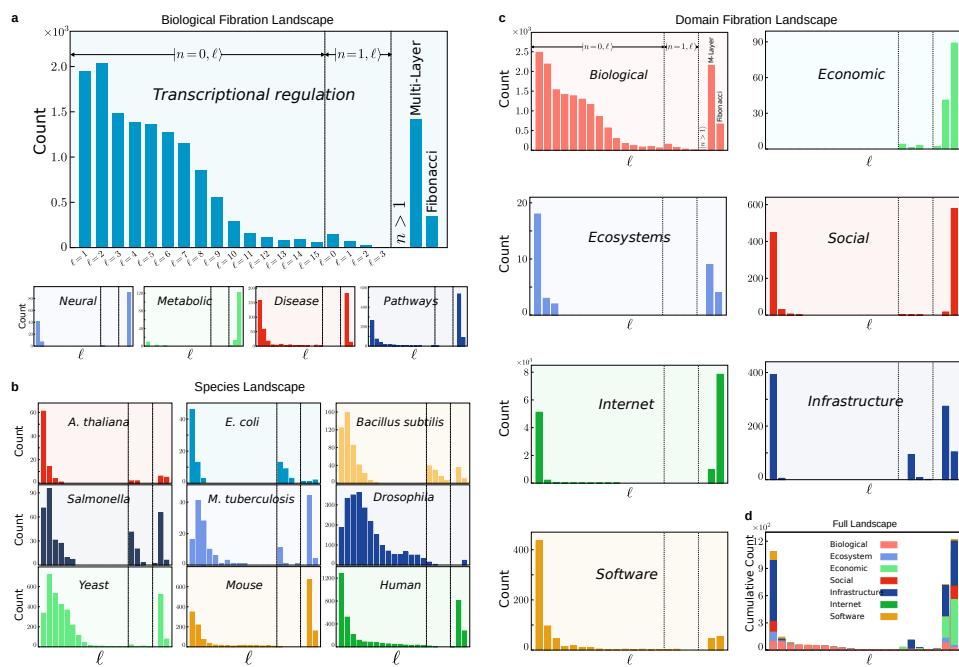


Fig. 15.16

Fibration building block landscape. (a) Distributions of building block fiber numbers across different types of biological networks, summed over the species. (b) Distributions of building block fiber numbers across species, from bacteria to humans, summed over the type of the network. (c) Distributions of building block fiber numbers across distinct domains of networks. (d) Distributions of building block fiber numbers for the union of all the domains in (c) normalized by the total number of nodes in the given domain. Results are scaled by multiplying by 10^4 . Figure reproduced from (Morone et al., 2020). Copyright © 2020, Leifer *et al.*

The complexity of fibration building blocks is intertwined with the complexity of their cycles. These cycles can be within the fiber, between the fiber and its regulators, or between fibers. The simpler building blocks are found in the TRN of *E. coli*. These building blocks are characterized by integer branching ratios $r = 0, 1, 2$. A few more complex (yet still simple) Fibonacci building blocks with fractal branching ratios are also found in *E. coli*. TRNs from more complex species show a high predominance of simple Fibonaccis. The level of complexity explodes when we look at metabolic networks. Thanks to intricate cycles, these networks are characterized by highly complex fibration symmetries and building blocks.

16.1 Metabolic networks

The fibration analysis of Chapter 15 studies building blocks in the TRN, mainly in the bacterium *E. coli*, but also in more complex eukaryotes. Since other network structures also exist, we focus next on metabolic networks. A *metabolic network* is composed of the chemical reactions of metabolic pathways, where enzymes transform metabolites that determine the physiological and biochemical properties of a cell. Figure 16.1 shows a typical construction of a metabolic network from its reactions, enzymes, metabolic products, and educts. In the most typical case, the metabolic network comprises metabolites as nodes and reactions as edges (Fig. 16.1a). This network representation does not capture the ODEs describing the dynamics of the metabolites. The hypergraph representation of Fig. 9.4b in Section 9.3 captures the correct representation of the ODEs for the fibration analysis of the metabolic network. A fibration analysis of these networks in *E. coli* reveals a just simple structure of building blocks.

However, by constructing a ‘dual’ network where the enzymes are the nodes and the edges are metabolites, a myriad of complex building blocks are revealed. These networks are called *enzyme networks*. In these networks, directed links resemble the transmission of metabolic information from an enzyme source to target enzymes. Therefore, enzyme networks serve as natural extensions of the TRNs discussed in the previous chapter, with transcription factors acting as messengers.

Such a view follows a distributed computational networked system captured by the fibration, where the behavior of network nodes is fiberwise constant; that is, processors in the same fiber are in the same state. Analogously, enzymes in fibers of the enzyme network are synchronous in a way akin to processing units that are synchronous in a computer.

Such an assumption suggests that enzyme-based fibers are elementary building blocks that bundle metabolic information in metabolic pathways.

Below, we provide a fibration analysis of enzyme networks following (Álvarez-García et al., 2025b) that uncovers building blocks of much greater complexity than those in TRNs. Compared to the *E. coli* TRN—which is mostly dominated by simple fiber structures—more complex structures exist in the metabolism, thanks to the intricate cycle structures of their enzyme pathways. These cycles are made of feed-forward and feedback structures that provide a systematic classification in terms of the fractal dimensions of the input trees.

These building blocks are more functionally homogeneous than enzymes in network motifs or network modules, pointing to a level of complexity in the organization and architecture of biological networks that topological motifs and modules found through statistical means simply miss. As a consequence, such fibration building blocks harbor more functional information compared to motifs and modules. Furthermore, since fibers represent information flow that goes beyond the boundaries of statistical over-representation of interactions between sets of nodes, such groups of enzymes may well be a better entry point not only to elucidate pathways from a different angle, but also to find novel pathways. Along the same lines, such fibers may also be used to find new drug targets, since fibers capture information flow, potentially providing a novel way to indicate points of therapeutic intervention.

16.2 Enzyme network of *E. coli*

Álvarez-García et al. (2025b) search for symmetries in the *E. coli* metabolic network built from the Ecocyc (Keseler et al., 2021) using the 'All Enzymes of E. coli K-12 Substrate MG1655' data set, together with information on metabolic reactions (Santos-Zavaleta et al., 2018) that are catalyzed by such enzymes from RegulonDB (Gama-Castro et al., 2016). These datasets provide metabolic reactions and corresponding enzymes, capturing 2,628 reactions between 2,093 metabolites. To curb the influence of ubiquitous metabolites such as ATP or H₂O, we sort all metabolites according to their occurrence in the underlying metabolic reactions and manually discard the most occurring metabolites. After this manual curation, we obtain 2,628 reactions with 1,990 nontrivial metabolites, catalyzed through 1,930 enzymes. To construct an enzyme-specific network (Fig. 16.1a), we connect enzymes when the product in a preceding reaction turns into a reactant in the subsequent reaction. As a result, we obtain an enzyme-enzyme network that captures 1,753 enzymes in a web of 22,011 directed edges.

16.3 Fibration analysis of the *E. coli* enzyme network

The first step in the analysis is to derive the SCCs of the network. The full enzyme network of 1,753 enzymes is characterized by a large SCC comprising 70% of the nodes. The other

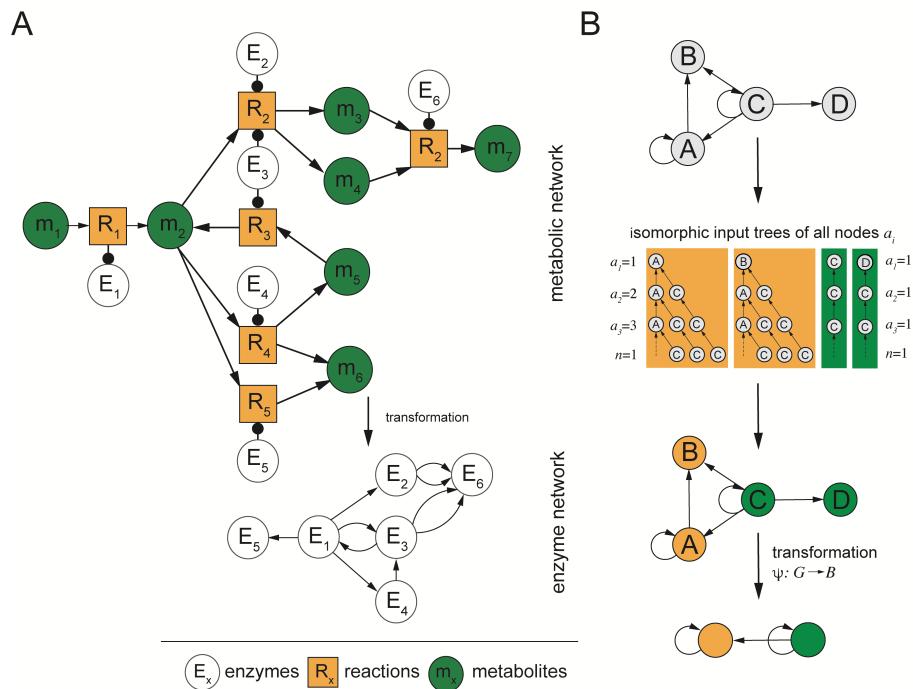


Fig. 16.1

Construction of enzyme metabolic networks and determination of symmetric network characteristics. (a) Enzyme network: Enzymes (E) are connected via a directed link when the products (m) of a preceding reaction (R) that is catalyzed by an enzyme are turned into a reactant in the subsequent reaction that is controlled through a different enzyme. (b) Using such directed enzyme networks, we perform fibration analysis in search of the building blocks. Figure reproduced from (Álvarez-García et al., 2025b).

SCCs are very small, composed of 4 enzymes on average. This is in contrast to the TRN, which contains a few small SCCs of similar size.

Four function-specific metabolic subnetworks are then considered. The relevance of metabolic reactions is established by the presence of transcription factors that control the expression of the corresponding enzymes, according to RegulonDB (Santos-Zavaleta et al., 2018). In particular, we filter metabolic reactions through enzymes that are regulated by transcription factors expressed during oxidative stress (Gama-Castro et al., 2014). In the same way, we extract transcription associated with carbon sources from the RegulonDB Sensor Units Datasets (Santos-Zavaleta et al., 2018). We extract carbon metabolic pathways where one-carbon moieties are transferred from donors to intermediate carriers, ultimately used in methylation reactions or in the synthesis of purine and thymidine, which are used in DNA building blocks. We also derive metabolic reactions of amino-acid metabolism from genes related to a specific growth condition obtained from (Martínez-Antonio et al., 2003). We also account for oxidative stress as a natural consequence of aerobic metabolism.

Finally, a glycolysis network is obtained by filtering all metabolic reactions catalyzed by enzymes that appeared in the corresponding KEGG pathways (Kanehisa et al., 2023). This is the major bacterial energy resource.

The fibers in these networks are obtained using the minimal balanced coloring algorithm of Section 13.7. We find 44 fibers in the amino-acid synthesis network, 32 fibers in the carbon metabolism network, 35 fibers in the glycolysis network (Fig. 16.2a) and 30 fibers in the oxidative stress network. The lists of all fibers of each network can be found in (Álvarez-García et al., 2025b). Fibers cover the networks significantly: the majority of nodes appear in fibers (Table 16.1). Moreover, at least roughly half of the nodes belong to an SCC, while a large fraction of such nodes appear in fibers as well.

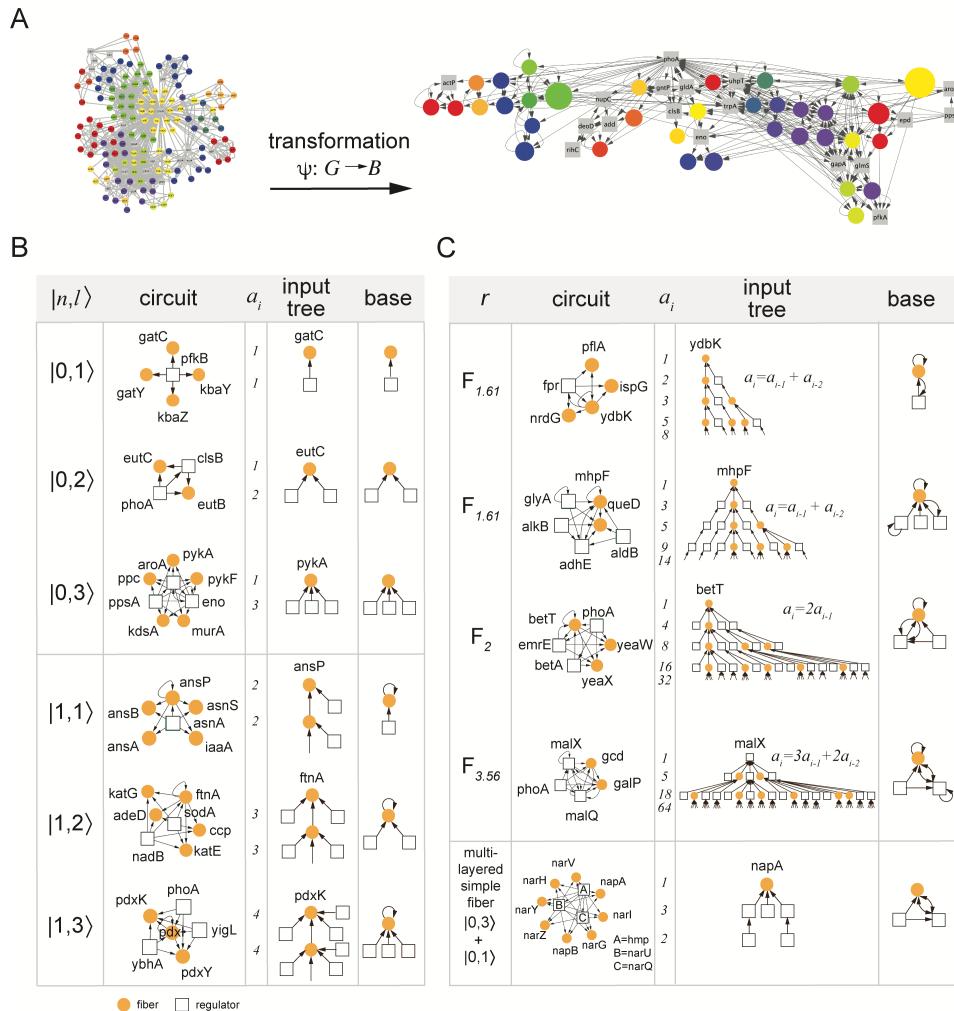
	<i>Amino Acid Synthesis</i>		<i>Carbon Metabolism</i>	
	<i>Nodes</i>	<i>%</i>	<i>Nodes</i>	<i>%</i>
Total nodes	261	100%	149	100%
Nodes in fibers	198	75.9%	95	63.8%
In SCCs (in fibers)	147 (107)	56.3(41)%	93 (65)	62.4(43.6)%
Connectors (in fibers)	18 (5)	6.9(1.9)%	22 (4)	14.8(2.7)%
k_{out} shell (in fibers)	96 (86)	36.8(33)%	34 (26)	22.8(17.4)%

	<i>Glycolysis</i>		<i>Oxidative Stress</i>	
	<i>Nodes</i>	<i>%</i>	<i>Nodes</i>	<i>%</i>
Total nodes	153	100%	168	100%
Nodes in fibers	135	88.2%	137	81.5%
In SCCs (in fibers)	80 (66)	52.3(43.1)%	80 (62)	47.6(36.9)%
Connectors (in fibers)	1 (0)	0.7(0)%	9 (3)	5.4(1.8)%
k_{out} shell (in fibers)	72 (69)	47.1(45.1)%	79 (72)	47(42.9)%

Table 16.1: **Structural composition of the four enzyme subnetworks.** We show the coverage of each network by fibers as well as the structural breakdown of the network. Typically, the networks consist of 4 SCCs, where one component captures the majority of nodes. Enzymes are also classified as connector nodes, or nodes in the k_{ou} shell, sending signals to other parts of the cell. Connector nodes send outputs to the SCCs, while the nodes in the k_{out} shell receive information only from nodes in the SCCs. Table reproduced from (Álvarez-García et al., 2025b).

16.4 Complex fibration building blocks

A first analysis reveals that the simple fibration building blocks found in the TRN in Chapter 15 are still present in enzyme networks (Fig. 16.2b). These include all fibers with integer branching ratio $n = 0, 1, 2$, the simple Fibonaccis, and the multilayer simple fibers. Their bases are sketched in the first three circuits in Fig. 16.3a.

**Fig. 16.2**

Fibers in the enzyme network. (a) Fibration applied to the glycolysis enzyme network in *E. coli*, where colors refer to fibers. (b) Simple fibration building blocks similar to those of the TRN are found in amino-acid synthesis, carbon metabolism, glycolysis, and oxidative stress enzyme networks. (c) Examples of complex Fibonacci building blocks. Figure reproduced from (Álvarez-García et al., 2025b).

However, these circuits form a minority of the fiber structures encountered: most fiber structures in enzyme networks correspond to much more complex structures. In particular, a pair of enzymes can ‘communicate’ through numerous different metabolites, so there are multiple different edges between the enzyme nodes. Such links lead to a more densely connected network structure, which produces a high number of cycles between nodes. However, these cycles appear not only between nodes in the same fiber but also between

nodes in different fibers. Such a feature is ubiquitous in metabolic networks. It produces much more complex structures, notably the Composite Fibonacci Fibration building block.

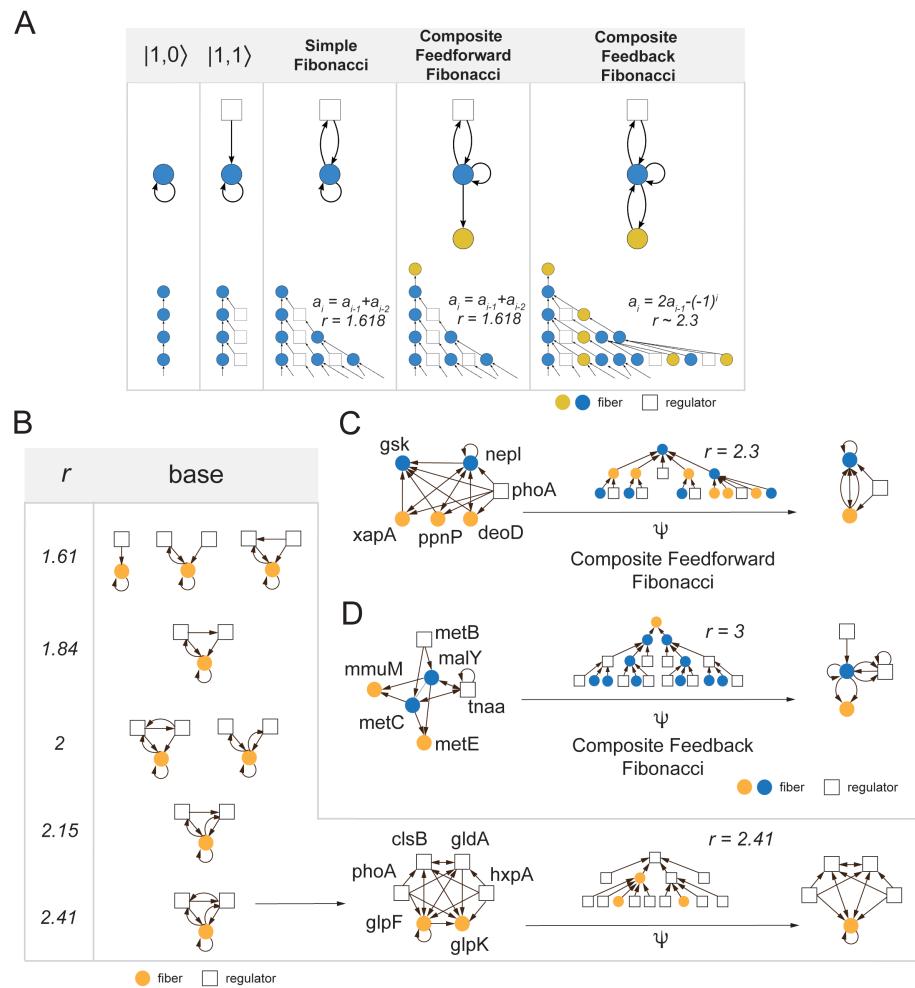
16.5 Composite Fibonacci fibration building blocks

A key structure in enzyme networks is the *Composite Fibonacci* building block composed of more than one fiber (Fig. 16.3a, c, d). These fibers are further split into *Composite feed-forward Fibonaccis* and *Composite Feedback Fibonaccis*. These building blocks are composed of either through a fiber that regulates another one in a feed-forward manner or by forming a feedback loop between the underlying fibers. Composite feed-forward Fibonaccis are instances of multi-layer fibers, in which a fiber regulates another in a feed-forward manner, where the regulating fiber belongs to a Fibonacci building block. In particular, the regulated fiber acquires a branching input tree from a regulating Fibonacci fiber, allowing its own input tree to branch. In Composite Feedback Fibonaccis, a layer of complexity is added by forming a loop between the fibers, such that each fiber acts as a regulator of the other.

These structures are abundant and can be systematically classified through their complexity by two ‘complexity axes’, facilitating the structured organization of building blocks according to their complexity levels. A ‘horizontal axis’ relates the simple building blocks to more complex types (Fig. 16.3a). Traversing different bases, we start with the base $|n = 1, l = 0\rangle$, a single fiber regulating itself. The addition of an external regulator gives the structure $|n = 1, l = 1\rangle$. Although both of these input trees are infinite as a result of self-regulation, they do not branch (Fig. 16.3a). However, when we add an edge from the fiber back to the regulator, we obtain the simplest Fibonacci fiber with a branching input tree with a fractal dimension of $r = 1.618$.

Adding a feedback loop between a regulator and a fiber that points to a simple Fibonacci fiber lets the corresponding input tree branch, gives a *Composite feed-forward Fibonacci*, since the Fibonacci fiber is now acting as a regulator of another fiber downstream. We show this case in Fig. 16.3a, where we add a yellow fiber that is regulated by a simple Fibonacci fiber in blue, creating a Composite feed-forward Fibonacci fiber. The input tree of the yellow fiber is given by the input tree of the blue fiber plus the new yellow node that connects to the root of the input tree of the blue fiber. Since the only difference between these input trees is the extra yellow node, the branching ratio remains unchanged from that of the simple Fibonacci and Composite feed-forward Fibonacci fibers. In turn, the addition of yet another edge that connects the new fiber with the original Fibonacci fiber creates a feedback loop, leading to a *Composite Feedback Fiber*. In consequence, the underlying input tree changes, leading to a branching ratio of 2.

A second ‘vertical axis of complexity’ corresponds to an increase in complexity through an increasing branching ratio. In Fig. 16.3b we increase the branching ratio from the simplest Fibonacci ($r = 1.618$) to a building block with a branching ratio of $r = 2.41$ in a Fibonacci building block that revolves around the regulation of *glpFK* through regulators that are connected through mutual feedback loops.

**Fig. 16.3**

Complexity axes of Fibonacci fibers. (a) ‘Horizontal axis’ shows transition from simple to complex types of building blocks through the addition of feedback loops. (b) ‘Vertical axis of complexity’ corresponds to an increase in complexity, which increases the branching ratio. Such structures emerge when more feedback loops between the underlying nodes are added. (c) In a feed-forward Fibonacci, we find a fiber *mmuM-metE* with a non-zero branching ratio even though it does not send any feedback to its regulators because it is regulated by another fiber *metC-malY*. (d) In a Feedback Fibonacci, two fibers (blue and yellow nodes) regulate each other, pointing to a branching input tree structure with $r = 2.3$. Figure reproduced from (Álvarez-García et al., 2025b).

In Fig. 16.3c, since the blue regulator fiber is part of a Fibonacci structure, the input tree of fiber *mmuM-metE* includes the input tree for *malY-metC*. This is a branching tree, making its input tree branch.

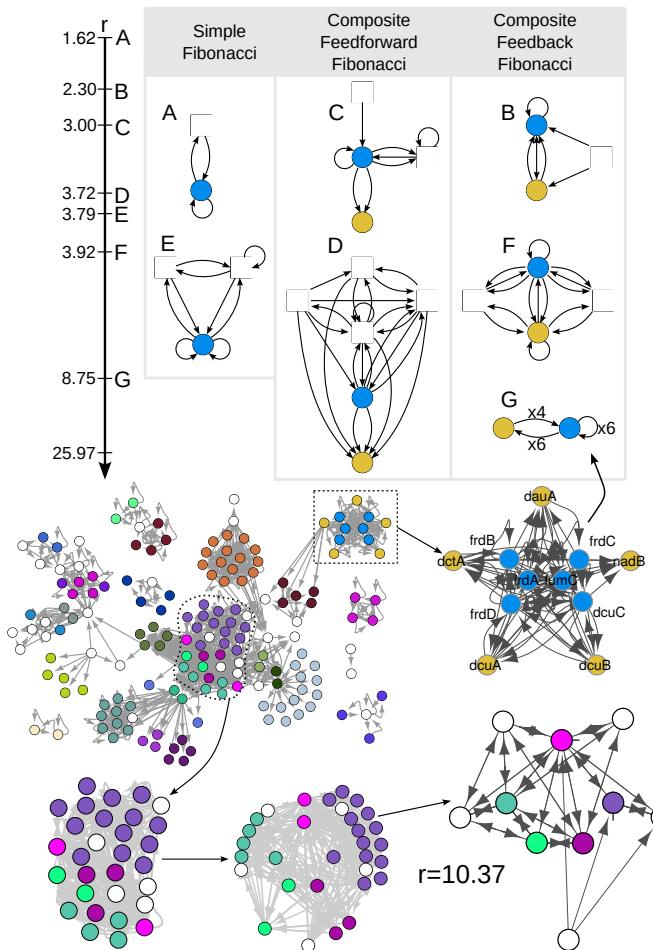


Fig. 16.4

Complexity ladder of the branching ratio. We show the bases of different building blocks corresponding to different types of *Fibonacci*. The axis on the left shows the range of branching ratios observed in the four different enzyme networks; labels correspond to the building blocks in the table. The first column depicts *Simple* *Fibonacci*s. The second column shows three different *Feedback* *Fibonacci*s. The third column depicts *feed-forward* *Fibonacci*s, where the fiber in a *Fibonacci* structure (blue fiber) regulates a second fiber (yellow fiber) in a feed-forward manner. The corresponding complete building block of structure **G** illustrates how such a complex structure can be reduced to a much simpler, more comprehensible structure at its base. Figure reproduced from (Álvarez-García et al., 2025b).

The type of *Fibonacci* structure in Fig. 16.3d features two (or even more) fibers representing a further step in complexity from previously observed fibers.

All *Fibonacci* structures, such as simple *Fibonacci*s, Composite feed-forward and Composite Feedback *Fibonacci*s are characterized by a branching input tree. Specifying only

the branching ratio of a building block does not determine the class of the building block. Two building blocks with the same branching ratio can belong to different classes, while a simpler class can have a higher branching ratio than a more complex class. For example in Fig. 16.4, building block E corresponds to a simple Fibonacci building block yet has a higher branching ratio $r = 3.79$ than blocks B and C, even though both are examples of Composite Fibonacci building blocks.

One way to determine the branching ratio of any input tree r , which is a simple practical option in the cases we need, is to determine the first few terms of the sequence a_i , which is the number of nodes in the i th layer of the input tree. This can be done by hand for some simple networks, using a recurrence relation such as the usual one for Fibonacci numbers if necessary. Then we compute the ratios $r_i = a_{i+1}/a_i$ and stop when the value of r_i , to the required accuracy, doesn't change (Álvarez-García et al., 2025b).

In general, however, there are two mathematical fine points. First, the sequence a_{i+1}/a_i may converge slowly. Second—although it is unlikely to arise in most applications—the sequence need not converge at all, as discussed in Section 14.5, so (14.3) must be used instead. In all cases the Perron–Frobenius Theorem implies that the branching ratio for a given node k , defined via (14.3), is the largest eigenvalue of the adjacency matrix for the induced subnetwork defined by all nodes that occur in the input tree of k . For a proof, which is not entirely straightforward, see (Boldi and Stewart, 2025). This eigenvalue can be computed rapidly by standard computer algebra packages.

In the next section, we analyze these complex building blocks in more detail.

16.5.1 Composite Feed-Forward Fibonacci building blocks

A Composite Feed-Forward Fibonacci is a multi-layer building block with a branching input tree, the key feature of Fibonacci fibers. Such a structure occurs when a fiber in a Fibonacci structure regulates (an)other fiber(s) downstream in a feed-forward manner. The Fibonacci fiber-regulator, as part of a Fibonacci structure, has a branching input tree that, in turn, gets ‘passed along’ to the nodes it regulates. In other words, the Fibonacci fiber ‘inherits’ and passes its input tree to the regulated fiber, causing the regulated fiber’s input tree to branch. This is exemplified in Fig. 16.3c where yellow nodes *mmuM* and *metE* only receive input from both *metC* and *malY*. Focusing on *metC* and *malY* we observe that both nodes are embedded in a Fibonacci structure through their regulator *tnaa*. As for the yellow fiber *mmuM* and *metE*, the input tree of the blue fiber is ‘inherited’, thus prompting its own input tree to branch as well.

A Composite Feedback Fibonacci occurs when the feedback loop (or loops) of a Fibonacci structure crosses two or more fibers, as in Fig. 16.3d. The input tree of the blue fiber *gsk-nepI* includes the nodes in the yellow fiber *xapA-ppnP-deoD* as well, since they act as regulators of the blue fiber. In turn, the nodes in the blue fiber regulate the yellow fiber, thus creating a loop between them. As a consequence of the loops between these fibers, the blue fiber also acts as a regulator to the yellow fiber. This loop causes the input tree to branch as $a_i = 1, 5, 11, 26, 137, 314\dots$, pointing to a branching ratio of $r = 2.3$.

As a consequence, one fiber can not be defined without the other(s), since each fiber acts as a regulator for the other fiber(s), suggesting that the separation of the regulator from

the other fiber is impossible. These structures can connect multiple fibers. In the simplest possible case, multiple fibers are linked in a ring-like cycle where each fiber is connected to only two others (*i.e.* one by an input and one by an output). In turn, all the fibers can be directly linked to each other, resembling an almost fully connected graph. Distinguishing these two structures, only one feedback loop (*i.e.* one single cycle of length n) encompasses all fibers in the first case, while a multitude of different feedback loops exists in the other case. In more detail, in a base that connects n fibers in such a fully connected scheme the total number of cycles comprises $\binom{n}{2}$ cycles of length 2 plus $\binom{n}{3}$ cycles of length 3, and so on up to one cycle of length n (without accounting for external regulators), pointing to a higher branching ratio.

The oxidative enzyme network in Fig. 16.4 provides examples of this kind of building block: the base of the structure shown at the bottom right corner is almost a fully-connected network encompassing 5 different fibers and 4 regulators that participate in the feedback loops (plus another regulator not included in any loop). We find a complex structure, pointing to an ‘entanglement’ of numerous cycles connecting several fibers with a high branching ratio. All enzyme networks exhibit this type of arrangement at the center of the network, as shown in Fig. 16.4 in the oxidative network. Such structures present the highest branching ratios. The amino-acid network has one such structure with $r = 12.39$, and another with $r = 25.97$, while the carbon network with $r = 10.81$, the glycolysis network has one with $r = 13.65$, and the oxidative network has one with $r = 10.37$.

As a corollary, these two composite Fibonacci structures can be combined, when a fiber in a Composite Feedback loop further regulates—albeit in a feed-forward manner—other fiber(s), as shown with the fibers regulated outwardly from the central Composite Feedback structure in the oxidative network at the bottom of Fig. 16.4.

Lastly, considering only *shortest* cycles to construct building blocks leads to a fiber that might be part of more than one building block. Such a situation occurs when longer cycles that cross (an)other fiber(s) are not included in a fiber’s building block, because they are longer than the shortest cycle between fibers and regulators. For example, the addition of a second fiber to a simple Fibonacci building block ($r = 1.618$) regulates the first fiber and the only external regulator. This addition creates a new cycle of length 3 that includes both fibers. When the building block of the original fiber is constructed, this new cycle is ignored since it is longer than the shortest cycle between the original fiber and its external regulator. However, when constructing the building block of the second fiber, the original fiber must be included since it regulates this newly added fiber. This second building block does include the longer cycle and both fibers, resulting in a Composite Feedback Fibonacci building block. This is observed numerous times in enzyme networks.

In such cases we consider the smaller building block to be the ‘defining’ one for the fibers included in both, since the shortest cycles have a dominant effect on the dynamics of this fiber or fibers. The bigger building block is naturally the defining one for only the fiber(s) included in it, and it can be considered a form of ‘higher order’ correction on the dynamics of the fiber(s) present in both building blocks.

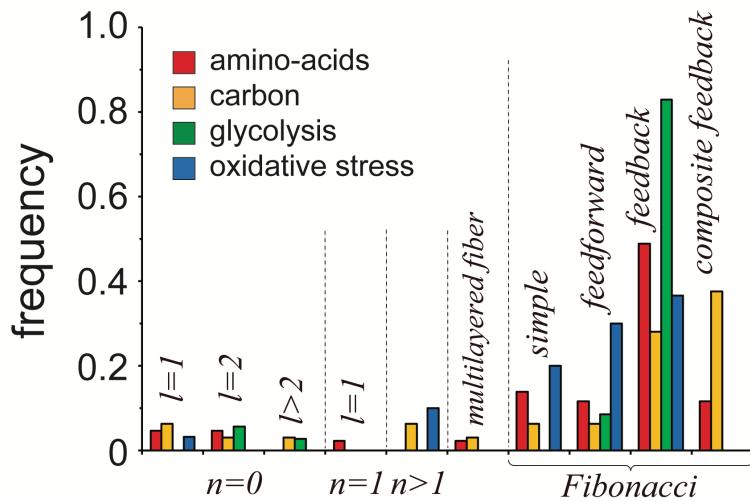


Fig. 16.5

Fiber landscape in the enzyme networks. Distribution of fibers in the amino-acid synthesis, carbon metabolism, glycolysis and oxidative stress networks, suggesting that Fibonacci fibers are most abundant in all networks. Figure reproduced from (Álvarez-García et al., 2025b).

16.6 Fibration landscape of enzyme networks

The large range of branching ratios found in metabolic networks, ranging from $r = 1.618$ to $r = 25.97$, prompts us to understand how adding or changing cycles influences the complexity of the circuits. Figure 16.4 showcases different examples of observed single and complex composite Fibonacci building blocks and their place in such a complexity landscape. The input tree is characterized by the sequence a_i of enzymes in the i th generation. As for Fibonacci fibers, this sequence can often be described by a closed form recurrence relation, such as $a_i = a_{i-1} + a_{i-2}$, which is observed with most basic Fibonacci fibers. The first term (a_{i-1}) represents a self-loop, a cycle of length 1, while the second term (a_{i-2}) represents a cycle of length 2 between the fiber and the regulator. Generally, a cycle of length d contributes a term a_{i-d} to the recurrence relation.

To increase the branching ratio of a building block, the number of cycles needs to be increased, which can be achieved by changing the multiplicity of the edges in the structure. For example, the block in Fig. 16.4g has essentially the same basic structure as the simplest Fibonacci structure in Fig. 16.4a. However, the multiplicity of the edges is higher, indicated by 4 directed edges that connect the yellow fiber to the blue fiber. In turn, 6 directed edges link the blue fiber with the yellow fiber, while the self-loop also has multiplicity 6, resulting in a structure with a highly elevated branching ratio $r = 8.75$. Alternatively, we can add more nodes and edges to increase the number of cycles, as in Fig. 16.3b. When the added

nodes form a part of a fiber, the added cycles create a Composite Feedback Fibonacci, since they connect fibers through a feedback loop.

Many nodes in fibers belong to SCCs, a natural consequence of having both a high coverage of nodes that belong to fibers, as well as a large percentage of nodes in the network belonging to SCCs. These two facts create the perfect scenario for the existence of feedback loops connecting multiple fibers, which point to very complex Composite Feedback Fibonaccis with several fibers. We observe this case in the building block at the center of the oxidative enzyme network (Fig. 16.4 bottom), where we find 5 fibers and 4 regulators entangled in a myriad of cycles between them. Such structures present the biggest branching ratios through an elevated number of different cycles that connect multiple fibers to a set of regulators. For example, the building block at the center of the oxidative enzyme network (Fig. 16.4) exhibits a branching ratio $r = 10.37$.

The landscape of fibration building blocks in the studied metabolic networks is dominated by highly complex Fibonacci structures (Fig. 16.5) rather than the simple blocks of the TRN. Given that these enzyme networks have many nodes in the SCCs and so many more loops than in the TRN, we can expect interactions between fibers, pointing to Fibonaccis as the most predominant fiber structure in these networks. In the TRN, we observe only three Fibonaccis with a branching ratio ranging between 1.38 and 1.61, while in enzyme networks, we observe values almost as high as ~ 26 . Such an observation is mainly a consequence of the increased number of cycles in these networks, ensuring multiple different possible ways to establish cycles between the fibers, and drastically increasing their complexity.

What can we learn from this massive organization of biological complexity into fibration building blocks? As fibers represent information flow that goes beyond the boundaries of statistical over-representation of interactions between sets of nodes, such groups of enzymes may well be a better entry point to not only elucidate pathways from a different angle, but also to find novel pathways. Along the same lines, such fibers may also be used as a way to find new drug targets, as fibers capture information flow, potentially providing a novel way to indicate points of therapeutic intervention.

This chapter explores the biological and statistical significance, as well as the functionality, of fibration building blocks through standard Gene Ontology analyses and *p*-value statistics. We compare the functionality of fibration building blocks to other methods of network decomposition, such as network motifs and modules. We discuss the insights that fibrations provide that are not captured by these alternative approaches, and we highlight the advantages of using fibrations to identify building blocks. Furthermore, we explain why alternative metrics based on statistical over-representation of circuits do not effectively capture the characteristics of functional building blocks.

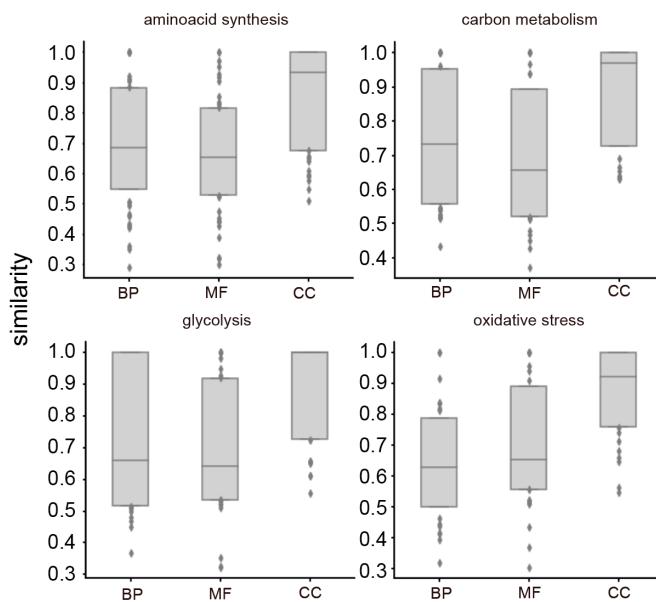
17.1 Biological significance of fibers

Chapters 15 and 16 have characterized the topological features of fibration building blocks as simple and complex. We have shown that increasingly larger cycle arrangements provide high complexity, but in a well-structured way that leads to a systematic classification in terms of cycles and branching ratios (Figs. 16.3, 16.4, 16.5). The crux of the matter, however, is to prove that these building blocks are significant for the functionality of the cell.

To this end, Álvarez-García et al. (2025b) consider the functional similarity of enzymes in the fibers obtained in Chapter 16, hypothesizing that participation in synchronized groups of enzymes translate into functional similarity. The mean similarity of Gene Ontology (GO) terms using GoSemSim (Yu et al., 2010) is determined over all pairs of enzymes in a fiber. The enrichment of a GO term is a guide to the biological function of any group of genes.

Fig. 17.1 shows the results for the enzyme networks studied in Chapter 16: amino-acid synthesis, carbon metabolism, glycolysis, and oxidative stress. We observe that enzymes in fibers are highly similar in their functions when we consider all three GO categories: biological processes, molecular function, and cellular components. As expected, when we include the regulators of the fibers in the analysis, they are not as functionally homogeneous as the enzymes within fibers are. This is because, most of the time, the regulators are master regulators participating in different fibers and, therefore, in different functions.

To compare with other notions of elementary building blocks of biological networks, we consider network motifs and modules, which are determined by statistical over-representation and network partitioning.

**Fig. 17.1**

Biological significance of fibers. Functional similarity of fibers in the amino-acid synthesis, carbon metabolism, glycolysis, and oxidative stress enzyme network. In each subnetwork, we calculated the mean similarities of all enzyme pairs in the fibers and their regulators. Fibers have high biological significance. Enzymes in fibers show high average functional similarity in the underlying four enzyme networks. In all Gene Ontologies including the molecular function (MF), biological processes (BP), and cellular components (CC) ontologies and subnetworks, we observe that enzymes in fibers are functionally highly similar. Figure reproduced from (Álvarez-García et al., 2025b).

17.2 Network motifs

The search for building blocks of biological networks is often motivated by the wish to find circuits that are easy to understand and occur frequently. They may also generate meaningful dynamics or perform signal processing tasks. Evolutionary pressures on gene expression dynamics may cause the evolution of suitable network structures: such structures will evolve and be conserved. They tend to accumulate as network motifs, or they may form symmetric structures such as fibers.

Are the symmetry-based building blocks the only way to decompose the network into fundamental elements? The answer is no. Meaningful patterns or building blocks may be defined in a number of ways. As hard-coded patterns (such as operons), statistically over-represented circuits (such as network motifs), circuits with specific simple shapes or symmetries (such as fibers), circuits forming modules, circuits performing specific dynamic

functions, or simply structures that are found to be evolutionarily conserved. Of course, these properties may also go hand in hand.

The most widely used way to define building blocks of biological and other networks are the motifs introduced by Alon and collaborators (Milo et al., 2002), which identify circuits of high statistical significance. These researchers pioneered the concept of a network motif as a statistical feature of the local topology of a graph. They were inspired by DNA sequence motif patterns, which are recurrent or conserved sequences of nucleotides with biological significance for the functioning of the cell, such as a binding site for a regulatory transcription factor. They observed that in biological networks, some simple motif network patterns appear more often than they would by pure chance.

Milo et al. (2002) suggest looking for patterns of connectivity in the network that occurs with a higher frequency than in random networks, in the following sense. They identify all possible subgraphs of size n , calculate the number of occurrences of each of them in the graph, and identify ones that occur significantly more often than in random networks generated by preserving the in- and out-degree for each node. They interpret this increased frequency as evidence that these motifs are basic building blocks for biological mechanisms. This proposal has had a major impact on systems biology (Alon, 2019; Klipp et al., 2016), and nowadays, finding motifs is a popular tool for analyzing network structure.

However, statistical abundance by itself does not imply that these circuits are fundamental bricks of biological systems. Fundamental circuits of biological networks can be underrepresented and appear only once in the network, yet they can have overwhelming importance for the functioning of the cell.

Motifs are good for finding basic building blocks that are statistically repeated ubiquitously. However, the functional significance of these network motifs has been debated, and results indicate that the link between motif and function does not exist (Ingram et al., 2006; Payne and Wagner, 2015; Macía et al., 2009; Ahnert and Fink, 2016). Indeed, except for the FAN motif, other motifs found by Alon and coworkers (Milo et al., 2002) do not lead to synchronization or other clear functional roles. Thus, the mere statistical significance of the network motif may not confer a definitive functionality within the cell machinery.

Another popular way to decompose a network is community detection (Girvan and Newman, 2002), also called network partitions or modularity (Hartwell et al., 1999a; Blondel et al., 2008). However, a biological module is not a partition of the network that is usually identified with a fundamental building block since many building blocks can form a larger module.

Fibration building blocks present a different approach: they identify the building blocks from a theoretical and principled approach through the theory of symmetry, rather than by statistics. Fibrations capture the topological properties, guaranteeing that a circuit can be organized into minimal forms of coherent function and logic computation.

By comparison, motifs cannot achieve these simple forms of coherent functionality: they do not synchronize nor oscillate, although admittedly, they were not designed to contribute toward these functional forms. The minimal canonical circuits that can achieve such behavior is related to the simple topological principle of symmetry invariance, which is necessary and sufficient to ensure that a network configuration can result in a coordinated function of gene dynamics, pointing towards a common functionality.

Below, we give a brief definition of network motif and modularity (community detection) algorithms, and then compare the topological decomposition and biological significance obtained by these methods to the decomposition by fibration building blocks. We explain why network motifs and community detection fail to find the fundamental bricks of biological networks.

17.2.1 Calculating network motifs

To find network motifs of size n in a graph G we can, in principle, take the following steps:

1. Identify all distinct topological types of subgraphs of size n : $G_1^n \dots G_m^n$ (we denote the total number of distinct graphs with n nodes by m).
2. Count the number of occurrences of G_i^n in G .
3. Assess the significance of each G_i^n .

The first two steps of this process are illustrated using Algorithms 6 and 7 below. The function $\text{CountMotifs}(G, G_1, \dots, G_m)$ counts the number of occurrences of motifs G_1, \dots, G_m in graph G . The function $\text{FindMotifs}(G, n)$ finds all possible motifs of size n in graph G and calculates their frequency of occurrence.

However, as the last part of this section explains, these two algorithms are seldom practical. On the one hand, there is a combinatorial explosion problem (line 1 of both algorithms); on the other hand, there is a graph-isomorphism test (line 4 of the first algorithm) for which there is no known polynomial-time algorithm.

Algorithm 6 : CountMotifs(G, G_1, \dots, G_m)

Input: Graph G , G_1, \dots, G_m (m subgraphs to look for).

Output: l_1, \dots, l_m (appearance count of each graph in $\{G_i\}$).

- 1: Identify all subgraphs of graph G : G^1, \dots, G^k
 - 2: $l_i = \text{array of } m \text{ integers equal to } 0$
 - 3: **for** $i \in [1, m]$ and $j \in [1, k]$ **do**
 - 4: **if** G_i is isomorphic to G^j **then**
 - 5: $l_i = l_i + 1$
 - 6: **end if**
 - 7: **end for**
 - 8: **return** $\{l_i\}$
-

Algorithm 7 : FindMotifs(G, n)

Input: Graph $G = (N, E)$, n - motif size of interest.

Output: All unique graphs with n nodes: G_1^n, \dots, G_m^n (m = number of unique graphs), l_1, \dots, l_m (occurrence frequency for each graph G_1^n, \dots, G_m^n).

- 1: Identify all unique graphs of size n : G_1^n, \dots, G_m^n
 - 2: $l_i = \text{CountMotifs}(G, G_1^n, \dots, G_m^n)$
 - 3: **return** $\{G_i^n\}, \{l_i\}$
-

To describe the third step, we consider, without loss of generality, a graph G and a single motif H . We define a function $f(G) = \text{CountMotifs}(G, H)$ that takes a graph G and a motif H and returns the number of motifs in G that are isomorphic to H . Consider now a distribution Y given by applying $f(G)$ to random graphs from the set $\{G^R\}$. Then Y can be found as $Y = f(G'), G' \in G^R$. The problem of defining the statistical significance of the motif splits into two steps: a) find the distribution Y of numbers of the motifs in random graphs, b) find the significance of the measurement $x = f(G)$ given the distribution Y .

The problem of finding the distribution of the occurrence frequency of a motif in a random graph does not have an analytical solution. Therefore, the distribution of Y cannot be found exactly. In practice, we can set up a computational experiment in which parameters of this distribution can be found (the distribution is assumed to be normal): M random graphs are generated preserving the in- and out-degree of each node in the original real network, and the number of occurrences of H is counted in each one of them ($\{l_i\}$). The Central Limit Theorem implies that for sufficiently large M the mean and standard deviation of Y can be estimated as the mean and standard deviation of $\{l_i\}$.

To discuss the significance of the measurement with the given distributions, we use the p -value and the Z -score. Studies of statistical significance are important not only to assess the significance of motifs but for any significance study of any building block. Explanations of how to calculate p -values and Z -scores can be found in any book on statistics. However, due to their importance for the present study, we now provide a succinct introduction of the basic concepts involved in using p -values to assess significance.

Due to the combinatorial nature of the number of motifs for a fixed n , i.e., the explosive factorial increase in the number of possible motifs with n , only values of n up to 5 are available. Then, for a fixed n , the number of occurrences of each subgraph is recorded and compared, using p -value statistics, with the numbers of occurrences in random networks.

17.2.2 Evaluating p -values and Z -scores

In statistics, p -values are used in the framework of hypothesis testing. The p -value is the probability of obtaining a measurement at least as extreme as the observed one, given that the *null hypothesis* is correct. The null hypothesis H_0 is a hypothesis to be ‘nullified’ in favor of an *alternative hypothesis* H_1 that is complementary to the null hypothesis. That is, the alternative hypothesis is considered valid and accepted if its complementary null hypothesis is exceedingly unlikely. The p -value is the probability of obtaining the observed measurement *given the null hypothesis*, which is different from the probability of the null hypothesis given the observed measurement; that is, $P(\text{measurement}|H_0) \neq P(H_0|\text{measurement})$.

In a standard problem in which we are given the measurement t and a statistic T , there are three possible ways to find the p -value. If we expect t to be bigger than T (one-sided right-tail test), the p -value (p_r) is defined as

$$p_r = P(T \geq t). \quad (17.1)$$

If we expect t to be smaller than T (one-sided left-tail test), the p -value (p_l) is defined as

$$p_l = P(T \leq t). \quad (17.2)$$

And if we expect t to be significantly different from T (two-sided test), then the p -value (p) is defined as

$$p = 2 \min(p_r, p_l). \quad (17.3)$$

The null hypothesis is rejected if the p -value is smaller than a threshold significance value α . A frequently used value of α in biology is 0.05, or even a less adventurous value of 0.1. These values should be compared to the p -values required in physics for a Nobel Prize-bound significant result. When the Higgs boson was discovered, the associated p -value was considered to be around 5σ , which translates to a probability of roughly 1 in 3.5 million, meaning a very small p -value of approximately 3×10^{-7} .

As an example, consider a coin tossing experiment. The null hypothesis H_0 is that the coin is fair ($q = 0.5$). The alternative hypothesis H_1 is that the coin is weighted towards heads ($q > 0.5$). Suppose the desired significance level is $\alpha = 0.05$. Suppose that the coin is tossed n times and m heads are observed. To find the p -value (p) of observing m heads in n tosses, assuming the coin is fair, we sum the probabilities of obtaining m or more heads using the formula:

$$p = \sum_{k=m}^{k=n} \frac{1}{2^n} \frac{n!}{k!(n-k)!}. \quad (17.4)$$

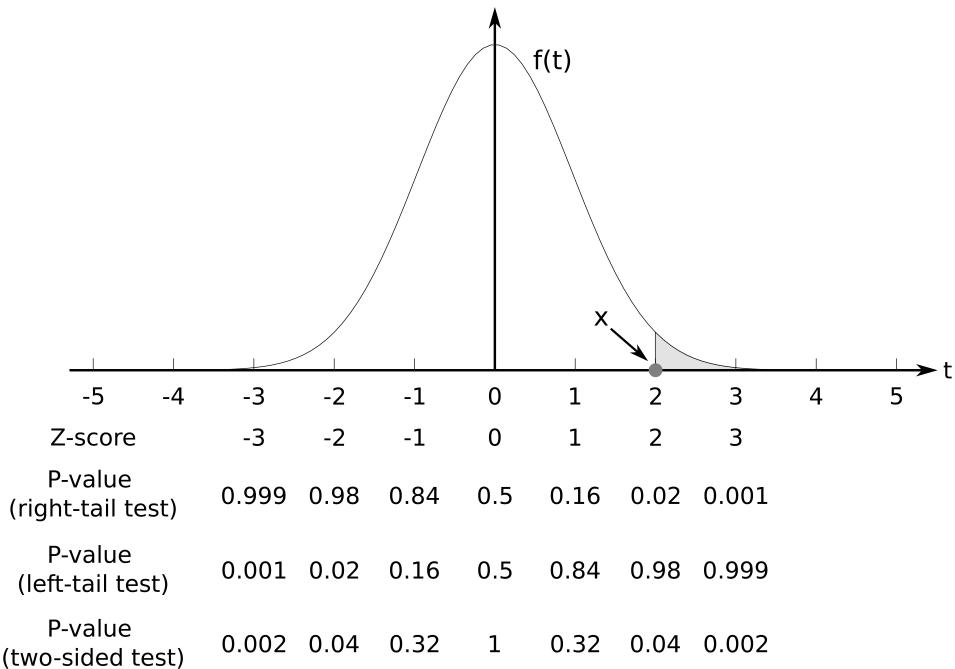
For instance, if $n = 10$ and $m = 9$, the p -value is

$$p = \sum_{k=9}^{k=10} \frac{1}{2^{10}} \frac{10!}{k!(10-k)!} = 0.010 + 0.001 = 0.011 \leq \alpha = 0.05. \quad (17.5)$$

This p -value is below the significance level of 0.05, so H_0 is deemed to be unlikely and is rejected. Then H_1 is accepted with a significance level of 0.05 (or 95%).

In this experiment, the probability distribution assumed for the null hypothesis is binomial (or its normal/Gaussian approximation), which is well supported by the experiment. It is important (but usually under-emphasized) that the null hypothesis tacitly involves assuming a specific probability distribution for the observations. Therefore, there can be two different reasons for ‘rejecting the null hypothesis’, which is the usual form of words used. One is its low probability, which in practice is seen as confirmation of the alternative hypothesis despite the careful phraseology of ‘rejecting the null hypothesis’. The other is that an inappropriate distribution is assumed when calculating the probability of such an extreme observation. Thus, it is vital to assume a plausible distribution. The default assumption of a normal distribution may not be appropriate despite the Central Limit Theorem.

When the distribution of the statistic T is normal, another metric is often used in combination with the p -value. The *Z-score* (also referred to as standard score) counts how far the measured value is from the mean of the distribution of the random value that the measured value is being compared to. That is, if the null hypothesis is that the measurement x belongs to a normal distribution with mean μ and standard deviation σ , then the *Z-score* (Z) of this

**Fig. 17.2**

Z-scores and p-values. Example calculation of Z-scores and p-values given the observation x (shown by the red dot) and the distribution $f(t) = N(0, 1)$. The p-value of x in the right-tail test is the probability of observing value at least as low as x and is given by $\int_x^\infty f(t) dt$ (area in cyan). Since the standard deviation of $f(t)$ is 1, Z-scores are equal to the value of t .

measurement is

$$z = \frac{x - \mu}{\sigma}. \quad (17.6)$$

Figure 17.2 illustrates the calculation of Z-scores and p-values given the measurement x and the normal distribution $f(t)$ with zero mean and standard deviation one.

17.2.3 Statistical significance of network motifs

To summarize, the significance of a motif H in a graph G can be found using the following steps:

1. Count the number of occurrences of H in G as $x = f(G)$.
2. Generate M random graphs G^1, \dots, G^M with the in- and out-degree of each node being equal to the in- and out-degree of the corresponding node in graph G .
3. Find mean $\mu(Y) = \langle f(G_i) \rangle$ and standard deviation $\sigma(Y) = \sqrt{\langle f(G_i)^2 \rangle - \langle f(G_i) \rangle^2}$.
4. Find Z-score as $z = \frac{x - \mu(Y)}{\sigma(Y)}$.

Pseudocode for finding network motifs of size n and estimating their significance is given

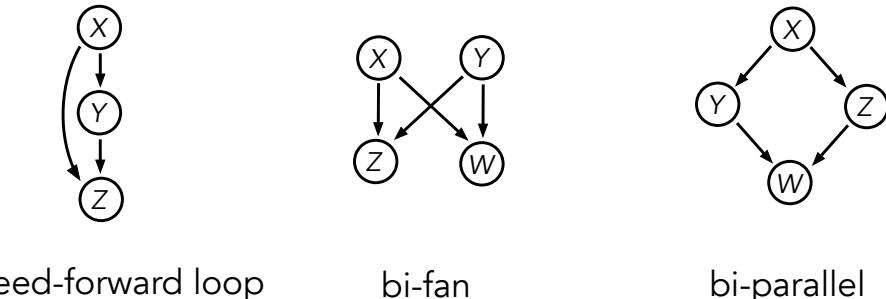


Fig. 17.3

Most common network motifs introduced by Milo et al. (2002). Feed-forward loop, bi-fan and bi-parallel.

in Algorithm 8. The function $\text{GetMotifs}(G, n)$ finds all unique motifs of size n in G , their number of occurrences in G , and the significance of each motif.

Algorithm 8 : GetMotifs(G, n)

Input: Graph $G = (N, E)$ (n = motif size of interest).

Output: All unique graphs with n nodes: G_1^n, \dots, G_m^n (m = number of unique graphs), occurrence frequency for each graph g_1^n, \dots, g_m^n and -score for each graph z_1^n, \dots, z_m^n .

- 1: Identify all unique graphs of size n : G_1^n, \dots, G_m^n
- 2: $g^n = \text{CountMotifs}(G, G_1^n, \dots, G_m^n)$
- 3: $l_{ij} = M$ by m array of integers equal to 0
- 4: **for** $i \in [1, M]$ **do**
- 5: Generate $G^i = (N_H, E_H)$ with $|N_H| = |N|$ and $|E_H| = |E|$ such that $k_{in}(n_j^H) = k_{in}(n_j^G)$ & $k_{out}(n_j^H) = k_{out}(n_j^G)$ for $j \in [1, |N|]$
- 6: $l_i = \text{CountMotifs}(G^i, G_1^n, \dots, G_m^n)$
- 7: **end for**
- 8: **for** $i \in [1, M]$ **do**
- 9: $z_i^n = \frac{g_i^n - \text{mean}(l_i)_j}{\text{sd}(l_i)_j}$
- 10: **end for**
- 11: **return** $\{G_i^n\}, \{g_i^n\}, \{z_i^n\}$

Here

$|N|$ = size of set N ,

k_{in} and k_{out} are the in- and out-degrees of a node,

$\text{mean}(l_i)_j$ and $\text{sd}(l_i)_j$ are the mean and standard deviation of the sample l_{ij} over the index j .

17.2.4 Network motifs in complex networks

Milo et al. (2002) apply the motif finding algorithm to a number of networks, including gene regulatory, neural, food webs, electronic circuits, and some others. Figure 17.3 shows

some of the most abundant motifs, and Fig. 17.4 shows the remaining ones across networks. One of the most abundant network motifs is the Feed Forward Loop (FFL) (Mangan and Alon, 2003), which we have already encountered in many discussions within this book. It is composed of a central Y gene that controls a Z gene, with another X gene controlling both, all in a feed-forward manner. Despite its name, there is no loop in this circuit. The functionality of the FFL is investigated in detail in Section 15.3. Briefly, it acts as a sign-sensitive delay element in transcription networks, delaying the signal reaching gene Z, and provides some protection from large short-lived fluctuations (Mangan et al., 2003).

As discussed in Section 15.3, the FFL is part of the FFF, which is obtained from the FFL by the addition of an AR loop at gene Y. The AR loop is another very common network motif. Here, we see one of the main problems with motif identification. Statistics alone will find these two circuits separately, but it will miss the main point that almost all of the time, these circuits appear together, forming the FFF. The FFF has the crucial property of synchronization, while its separate pieces, FFL and AR, have no coherent functionality, except that an FFL produces a delay in the signal arriving at Z by the intermediate step at Y. This has been proposed by (Mangan et al., 2003) to be useful since it protects gene Z from large rapid fluctuations in the inputs. While useful, this functionality does not seem to be of enough importance to declare the FFL as the quintessential motif of biological machineries.

The bi-fan motif in Fig. 17.3 is actually a fiber, which is found by fibration analysis as the $|n = 0, l = 2, m = 2\rangle$ fiber. It is a rather trivial fiber which produces synchronization in the Z and W genes by their shared inputs and in the TRN acts as a regulon, as discussed in Section 15.8. Finally, the bi-parallel motif is also part of a trivial fiber, in that the genes Y and Z are synchronized in a fiber by sharing the same input, as discussed in Section 15.8.

An important concept in this work is the distinction between synchronization resulting from trivially sharing the same inputs (as in operons and regulons) versus synchronization induced by a more complex symmetry fibration. That is, the difference between coregulation induced by a fiber (coexpression resulting from shared input trees, which take into account extended paths in the network) versus coregulation by a single input of a regulon. Both lead to coexpression, but the former is more complex than the latter, which we consider to be a trivial form of synchronization, unlike fiber synchronization.

17.2.5 Drawbacks of network motifs

The very high Z-scores obtained for the motifs (Milo et al., 2002) point to the conclusion that motifs do indeed appear more frequently in the studied networks, and therefore, are ‘the basic building blocks of complex networks’. While this deduction has had considerable influence in systems biology (Alon, 2019; Klipp et al., 2016), the functional role of motifs is still questionable (Ingram et al., 2006; Macía et al., 2009; Payne and Wagner, 2015; Ahnert and Fink, 2016).

From the dynamical standpoint, there is an obvious flaw in this logic. Motif dynamics studied in isolation can be very different from the dynamics of the same circuit when it is embedded in the network. This is because a motif is a subgraph, which ignores links to external nodes that can interfere with the motif dynamics. Thus, the isolated dynamics

network	nodes	edges	N_{real}	N_{rand} ± SD	Z-score	N_{real}	N_{rand} ± SD	Z-score	N_{real}	N_{rand} ± SD	Z-score
gene regulation (transcription)							feed-forward loop		bi-fan		
<i>E.coli</i>	424	519	40	7 ± 3	10	203	47 ± 12	13			
<i>S.cerevisiae</i>	685	1052	70	11 ± 4	14	1812	300 ± 40	41			
neurons							feed-forward loop		bi-fan		bi-parallel
<i>C.elegans</i>	252	509	125	90 ± 10	3.7	127	55 ± 13	5.3	227	35 ± 10	20
food webs							Three chain		bi-parallel		
Little Rock	92	984	3219	3120 ± 50	2.1	7295	2220 ± 210	25			
Ythan	83	391	1182	1020 ± 20	7.2	1357	230 ± 50	23			
St. Martin	42	205	469	450 ± 10	NS	382	130 ± 20	12			
Chesapeake	31	67	80	82 ± 4	NS	26	5 ± 2	8			
Coachella	29	243	279	235 ± 12	3.6	181	80 ± 20	5			
Skipwith	25	189	184	150 ± 7	5.5	397	80 ± 25	13			
B. Brook	25	104	181	130 ± 7	7.4	267	30 ± 7	32			
electronic circuits (forward logic chips)							feed-forward loop		bi-fan		bi-parallel
s15850	10,383	14,240	424	2 ± 2	285	1040	1 ± 1	1200	480	2 ± 1	335
s38584	20,717	34,204	413	10 ± 3	120	1739	6 ± 2	800	711	9 ± 2	320
s38417	23,843	33,661	612	3 ± 2	400	2404	1 ± 1	2550	531	2 ± 2	340
s9234	5,844	8,197	211	2 ± 1	140	754	1 ± 1	1050	209	1 ± 1	200
s13207	8,651	11,831	403	2 ± 1	225	4445	1 ± 1	4950	264	2 ± 1	200
electronic circuits (digital fractional multipliers)							three-node feedback loop		bi-fan		four-node feedback loop
s208	122	189	10	1 ± 1	9	4	1 ± 1	3.8	5	1 ± 1	5
s420	252	399	20	1 ± 1	18	10	1 ± 1	10	11	1 ± 1	11
s838	512	819	40	1 ± 1	38	22	1 ± 1	20	23	1 ± 1	25
world wide web							feedback with two mutual dyads		fully connected triad		uplinked mutual dyad
nd.edu	325,729	1.46e6	1.1e5	2e3 ± 1e2	800	6.8e6	5e4 ± 4e2	15,000	1.2e6	1e4 ± 2e2	5000

Fig. 17.4

Other network motifs in complex networks after (Milo et al., 2002). Tree chain, three-node feedback loop, four-node feedback loop, feedback with two mutual dyads, fully connected triad, and uplinked mutual dyad.

of the motif have nothing to do with the dynamics in the full network. This impedes any functional interpretation of the motif. Fibers, on the contrary, have the same synchronization properties when studied in isolation or in the network, as shown in the comparison between the FFL motif and the FFF in Fig. 17.5.

Thus, a network motif that is analyzed in isolation need not behave in the same manner when it has inputs from the rest of the network. Therefore, any conclusion about the phenotype of a network motif must be viewed with caution. In contrast, fibration building blocks are induced subgraphs that are designed in such a way that all the inputs to the fiber are included in the analyzed network so that the solution obtained for the fibers are the real solutions that would be encountered when the fiber is embedded in the full network.

Another important limitation is the size of the motif. The problem of finding all topologically distinct graphs of size n becomes computationally intractable even for fairly small n . We could ask whether, for instance, a complex fiber structure, such as any of the Fibonacci fibers can be ever found by statistical analysis alone, à la motifs. The answer would be ‘no’. This circuit is too large to be uncovered by statistical over-representation in a random search over all possible circuits of a given node size.

If we believe that each gene should play a significant role and has been evolutionarily optimized for that role, it makes sense to ask: How is a given node regulated overall, and which nodes share the same coherent patterns overall? Network fibers directly analyze the ‘history’ of input signals to a gene from its regulators, the regulators of its regulators, etc., with an ‘infinite history’ if the regulation involves loops. They define an equivalence between genes if their input histories look structurally the same and use this to split the expression network exhaustively into gene groups, such that—ideally—each group yields different expression patterns in response to the network’s input signals, but all genes in the same group shows the same expression pattern. The existence of fibers addresses functionality directly through synchronization.

Some fibers are also motifs, like the $n = 0$ class, which is analogous to a FAN motif or an autoregulation loop, but fibers admit more general forms that are related to the dynamical state of the genes and their function. Thus, fibers can play the role of building blocks, as can network motifs, but there are clear differences:

- Fibers are not defined (and revealed) by mere counting but by analyzing symmetry structures in the network.
- Looking at the FFL from the viewpoint of fibrations brings an interesting insight. Consider an isolated, coherent positive FFL as an example (genes denoted by X, Y, Z in Fig. 17.3). If the internal node Y contains a positive self-loop, then Y and Z form an FFF, so their expression is synchronized. As discussed, this internal self-regulation in FFLs is often observed and now acquires a natural explanation.
- In analyzing the dynamics and function of network motifs, we consider the motif in isolation. This works well for motifs in which signals propagate in only one direction, as in the FFL. But if the motif is embedded in a network, and the motif’s output signal feeds back into its input, this is considered ‘out of scope’. This explains why the motif dynamics need not persist when the motif is embedded in the full network. In gene

fibers, such infinite loops are not only handled properly but are even part of the very definition of the fiber.

These differences are illustrated in Fig. 17.5, which displays a sample network composed of FFFs and FFL motifs, along with an analysis of its fibration and network motifs building blocks. In Fig. 17.5b, we see that this network consists of 2 FFFs and 1 Fibonacci fiber building block, arranged into a multilayer fiber as depicted at the base. This analysis accurately represents the synchrony in the balanced coloring and shows no overlap among the fibers, except for the correct interactions between the regulators; one regulator can regulate multiple fibers. In contrast, the network motif analysis presented in Fig. 17.5c reveals 4 FFLs and 3 AR loops. None of these motifs are functional, nor are they significant for the dynamics of the circuit. In particular, two FFLs overlap; the Fibonacci fiber is formed by two overlapping FFLs.

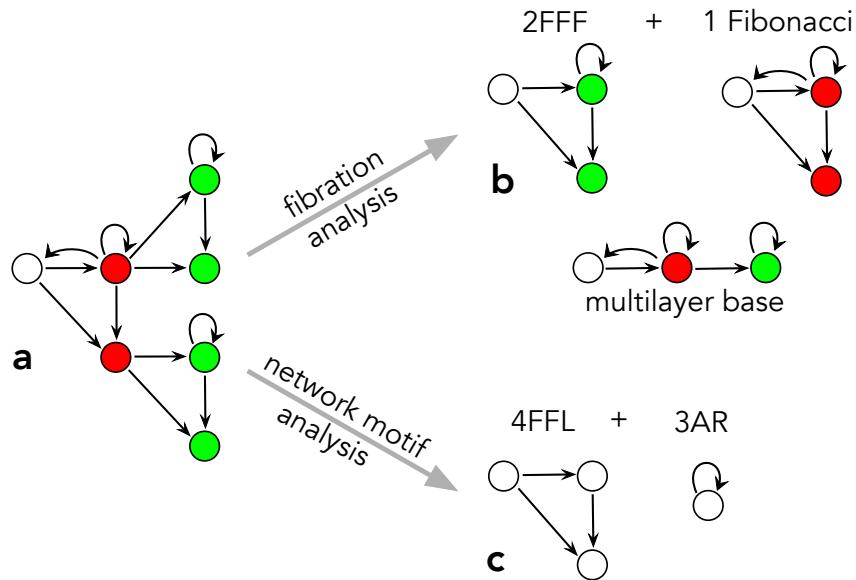
As remarked, the main problem with motifs is that edges connecting to the motif from the outside are not taken into account. These can change the dynamics, compared to how the motif would behave on its own. In contrast, fibers have, by definition, the same dynamical behavior, whether they are analyzed in isolation or embedded in the network. This might explain why the question of whether motifs have a functional role remains controversial (Ingram et al., 2006; Macía et al., 2009; Payne and Wagner, 2015; Ahnert and Fink, 2016). Fiber building blocks, on the contrary, are identified in the network structure in such a way that their dynamics are much the same whether they are studied in isolation or in the network.

Despite these drawbacks, motifs can still be a useful mathematical tool to decompose a network. However, caution is required when making any conclusion about the phenotype of a motif.

17.3 Statistical significance of fibration building blocks

Even though fibration building blocks are not found by statistical over-representation, it is important to understand, after finding them by a fibration analysis, whether they are statistically significant. That is, whether these circuits could equally probably have been generated by random addition of links to the network. We can guess, correctly, that the answer is ‘no’. Intuitively, intricate circuits such as Fibonacci and FFF are unlikely to be the result of randomness. While randomness plays an important role in evolution through mutations, evolutionary pressure will eventually lead to very structured circuits that cannot be compared to those obtained by random addition of edges without selection for functionality. Nevertheless, it is important to use the statistical analysis applied to motifs in Section 17.2.3 to confirm this intuition.

Leifer et al. (2020) study the statistical significance of the simple fibration building blocks discussed in Chapters 14 and 15 using the regulatory networks of a wide range of species spanning *A. thaliana*, *M. tuberculosis*, *B. subtilis*, *E. coli*, *salmonella*, yeast, mouse and

**Fig. 17.5**

Drawbacks of network motifs as building blocks. (a) Sample network with balanced coloring indicating fibration symmetry. (b) The sample network is composed of 2 FFFs and 1 Fibonacci fiber arranged into a multilayer base capturing the correct dynamics for the balanced coloring. (c) A motif analysis identifies 4 FFLs and 3 AR loops whose independent dynamics do not correspond to the dynamics embedded in the network.

humans. The datasets are described in Table 17.1. Leifer et al. (2020) also provide, in the supplementary material sections, a plot of every single building block found for all species and networks. We show in Table 17.2 the count of building blocks across species and their associated Z-scores; the large Z-scores show that these circuits are statistically significant.

17.4 Modularity

Modularity is a basic property of any network. It refers to the notion that there exists modules or clusters of nodes in a network with preferentially many links among themselves and with fewer links to nodes in other modules. In social networks, modularity is synonymous with community structure, and modularity and community detection algorithms have primarily been designed to uncover coherent groups in social networks. In these networks, it is expected that socially coherent groups should have many links in common and be weakly connected to the rest. Many algorithms have been designed to capture this modular structure in networks. There are excellent reviews covering the extensive literature on community and modularity detection algorithms (Fortunato, 2010); see also Chapter 9 of (Barabási and Pósfai, 2016) for an introduction to community detection and modularity. Here, we

mention just a few of the most popular methods. In Section 17.5, we discuss their use for identifying clusters. It should be noted that modularity applied to a functional network

Species Database	Additional information
Arabidopsis Thaliana ATRM (Jin et al., 2015)	We use high-confidence functionally confirmed transcriptional regulatory interactions from the ATRM database of the broadly used model plant Arabidopsis. http://atrm.cbi.pku.edu.cn/
Micobacterium Tuberculosis Research article (Balázsi et al., 2008)	Supplementary Information of (Balázsi et al., 2008) https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2600667/bin/msb200863-s2.xls
Bacillus subtilis SubtiWiki (Zhu and Stölke, 2018)	We download the database from the SubtiWiki website and consider all repressor and activation links as ‘Repression’ and ‘Activation’. This database is considered to be the primary source of information for Bacillus. http://subtiwiki.uni-goettingen.de/
Escherichia coli RegulonDB (Gama-Castro et al., 2016)	We use the TF - operon interaction network from (Gama-Castro et al., 2016). RegulonDB combines transcriptional regulator interactions obtained by curating literature and using NLP high-quality data and partially confirmed experimentally and computationally predicted data. http://regulondb.ccg.unam.mx/
Salmonella SL1344 SalmoNet (Métris et al., 2017)	We use the regulatory layer of the strain Salmonella Typhimurium SL1344. SalmoNet consists of manually curated low-throughput and high-throughput experiments and predictions based on experimentally verified binding sites and TF-gene binding site data from RegulonDB. http://salmonet.org/
Yeast YTRP (Yang et al., 2014)	We use the TF-gene regulatory and TF-gene binding networks. Results of the TFPEs (Transcription Factor Perturbation Experiments) identify the regulatory targets of TFs. This is further refined by using literature-curated data. http://cosbi3.ee.ncku.edu.tw/YTRP/Home
Mouse TRRUST (Han et al., 2017)	Downloaded from TRRUST website. TRRUST is constructed using sentence-based text mining of more than 20 million abstracts from research articles refined by manual curation. https://www.grnpedia.org/trrust/
Human TRRUST (Han et al., 2017) TRRUST_2 (Han et al., 2017) KEGG (Kanehisa et al., 2015)	Downloaded from TRRUST website. Downloaded from TRRUST website and curated. We use KEGG API to download all pathways of Human gene regulatory network. Then all networks are combined and duplicates are removed. https://www.genome.jp/kegg/pathway.html

Table 17.1: Description of biological dataset. We use these datasets of biological networks to assess the statistical significance of fibration building blocks. All data are gathered from the stated sources. Table reproduced from (Leifer et al., 2020).

Species	Database	Nodes	Edges	AR Fiber			FFF		
				N_{real}	$N_{\text{rand}} \pm SD$	Z-score	N_{real}	$N_{\text{rand}} \pm SD$	Z-score
Arabidopsis Thaliana	ATRM	790	1431	2	0.2 ± 0.5	4	2	0 ± 0	Inf
Micobacterium tuberculosis	Research article	1624	3212	11	0.7 ± 0.8	13.2	6	0.2 ± 0.4	14.6
Bacillus subtilis	SubtiWiki	1717	2609	35	0.3 ± 0.5	64.6	13	0.3 ± 0.5	23.4
Escherichia coli	RegulonDB	879	1835	14	0.2 ± 0.5	29.1	12	0.1 ± 0.2	49.4
Salmonella SL1344	SalmoNet	1622	2852	21	0.7 ± 0.8	25	14	0.2 ± 0.4	32
Yeast				10		5			
	YTRP.regulatory	3192	10947	10	0.3 ± 0.6	17.3	4	0.2 ± 0.4	8.5
Mouse	YTRP.binding	5123	38085	2	0.1 ± 0.3	6.3	0	N/A	N/A
	TRRUST	2456	7057	1	0.1 ± 0.4	2.3	0	N/A	N/A
Human				1		1			
	TRRUST	2718	8215	0	N/A	N/A	0	N/A	N/A
	TRRUST.2	2862	9396	0	N/A	N/A	0	N/A	N/A
KEGG		5164	59680	1	0.06 ± 0.25	3.76	1	0 ± 0	> 3
				100		1			
Species	Database	Nodes	Edges	Fibonacci fiber			$n = 2$ Fiber		
				N_{real}	$N_{\text{rand}} \pm SD$	Z-score	N_{real}	$N_{\text{rand}} \pm SD$	Z-score
Arabidopsis Thaliana	ATRM	790	1431	5	0.3 ± 0.6	8.1	0	N/A	N/A
Micobacterium tuberculosis	Research article	1624	3212	4	1.7 ± 1.4	1.7	0	N/A	N/A
Bacillus subtilis	SubtiWiki	1717	2609	1	1.3 ± 1.2	-0.2	2	0 ± 0	63.2
Escherichia coli	RegulonDB	879	1835	2	0.5 ± 0.8	1.9	1	0 ± 0	> 3
Salmonella SL1344	SalmoNet	1622	2852	2	1.4 ± 1.3	0.5	3	0 ± 0	> 3
Yeast				3		0			
	YTRP.regulatory	3192	10947	2	1.8 ± 1.3	0.2	0	N/A	N/A
Mouse	YTRP.binding	5123	38085	0	N/A	N/A	0	N/A	N/A
	TRRUST	2456	7057	6	0.3 ± 0.6	9.3	0	N/A	N/A
Human				100		1			
	TRRUST	2718	8215	10	0.4 ± 0.6	16.3	0	N/A	N/A
	TRRUST.2	2862	9396	11	0.4 ± 0.7	16	0	N/A	N/A
KEGG		5164	59680	79	0.6 ± 0.7	112	1	0 ± 0	> 3

Table 17.2: **Fibers are statistically significant over many networks.** We report the Z-scores showing that all fibers found are statistically significant. We use a random null model with the same degree sequence (and sign of interaction) as the original network to calculate the random count N_{rand} and compare this with the real circuit count N_{real} to get the Z-score. Table reproduced from (Leifer et al., 2020).

provided synchronized clusters, but applied to a structural network does not, more details in Section 23.5.4.

Louvain modularity and hierarchical clustering

Louvain modularity is a popular algorithm designed to identify communities or modules in networks (Blondel et al., 2008). These communities represent a partition of nodes into groups within which the network connections are dense but sparser otherwise.

Using a greedy agglomerative procedure, the Louvain clustering algorithm maximizes the modularity Q of a partition $C = C_1, \dots, C_p$ of a directed graph G (Fig. 17.6a):

$$Q = \frac{1}{m} \sum_{u,v} \left[E_{uv} - \frac{d_u^{\text{in}} d_v^{\text{out}}}{m} \right] \delta(C_u, C_v), \quad (17.7)$$

where $m = |E|$ is the number of edges of G , E_{uv} represents the existence (0 or 1) of an edge from community u to community v , $d_u^{\text{in/out}}$ represent in/out degrees of community u . There are many variations of this algorithm, including ones for undirected or weighted networks (Fortunato, 2010).

Another popular algorithm was introduced by Girvan and Newman (2002) for community detection in social networks. In the Girvan-Newman algorithm, the *edge betweenness centrality* (EBC) is defined to be the number of shortest paths that pass through a given

edge in a network. Every edge is assigned an EBC score based on the shortest paths among all the nodes in the graph.

Finding EBC scores is an iterative process: take one node at a time and plot the shortest paths to the other nodes from the selected node. Based on these shortest paths, compute the EBC scores for all the edges. Repeat this process for every node in the graph. Now, every edge gets n scores corresponding to n nodes in the graph. These scores are added edgewise.

The Girvan–Newman algorithm uncovers communities in a graph by iteratively removing the edges of the graph based on the EBC score. Edges with the highest score are removed until the graph splits into two. This constitutes one step of the algorithm.

Consider a particular division of the graph into k clusters (communities). Define a $k \times k$ symmetric matrix e , where the element e_{ij} is the fraction of all edges in the network that link nodes in community i to nodes in community j . Then the trace $\text{Tr } e = \sum_i e_{ii}$ gives the fraction of edges in the network that connects vertices in the same community. Clearly good division in communities should have a high value for this trace.

Define $a_i = \sum_j e_{ij}$, which represents the fraction of edges that connect to vertices in community i . The modularity of a graph is then defined as:

$$Q = \sum_i (e_{ii} - a_i^2) = \text{Tr } e - \|e^2\|, \quad (17.8)$$

where $\|x\|$ indicates the sum of the elements of the matrix x . The quantity Q measures the fraction of the edges in the network that connects vertices of the same type (i.e., within-community edges) minus the expected value of the same quantity in a network with the same community divisions but random connections between the vertices. The sequence of splits determines a dendrogram, as in Fig. 17.6b. Modularity is then calculated for every split of the network as we move down the dendrogram and look for local peaks in its value, which indicates particularly satisfactory splits.

Further popular methods to find communities and modules are hierarchical clustering algorithms using agglomerative and divisive methods (Fortunato, 2010). These methods also produce a hierarchical tree or dendrogram, illustrating output generated by the algorithm (Fig. 17.6b). The circles at the bottom represent individual nodes. These nodes are connected at a given level of the tree according to different strategies measuring similarity

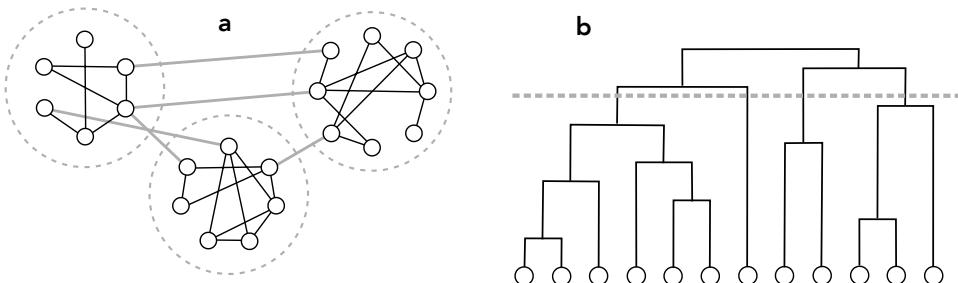


Fig. 17.6 Measurements of modularity. **(a)** Louvain-type modularity. **(b)** Hierarchical modularity.

between nodes. As we move up the tree, nodes form larger communities, culminating in a single community at the top. The dendrogram illustrates an initially connected network splitting into smaller communities. The resulting communities are subjective due to the uncertainty of where to cut the dendrogram.

These methods just scratch the surface of the large number of algorithms. The interested reader is referred to extensive reviews in the literature (Fortunato, 2010).

17.5 Comparison of fibrations, motifs, and modules in enzyme networks

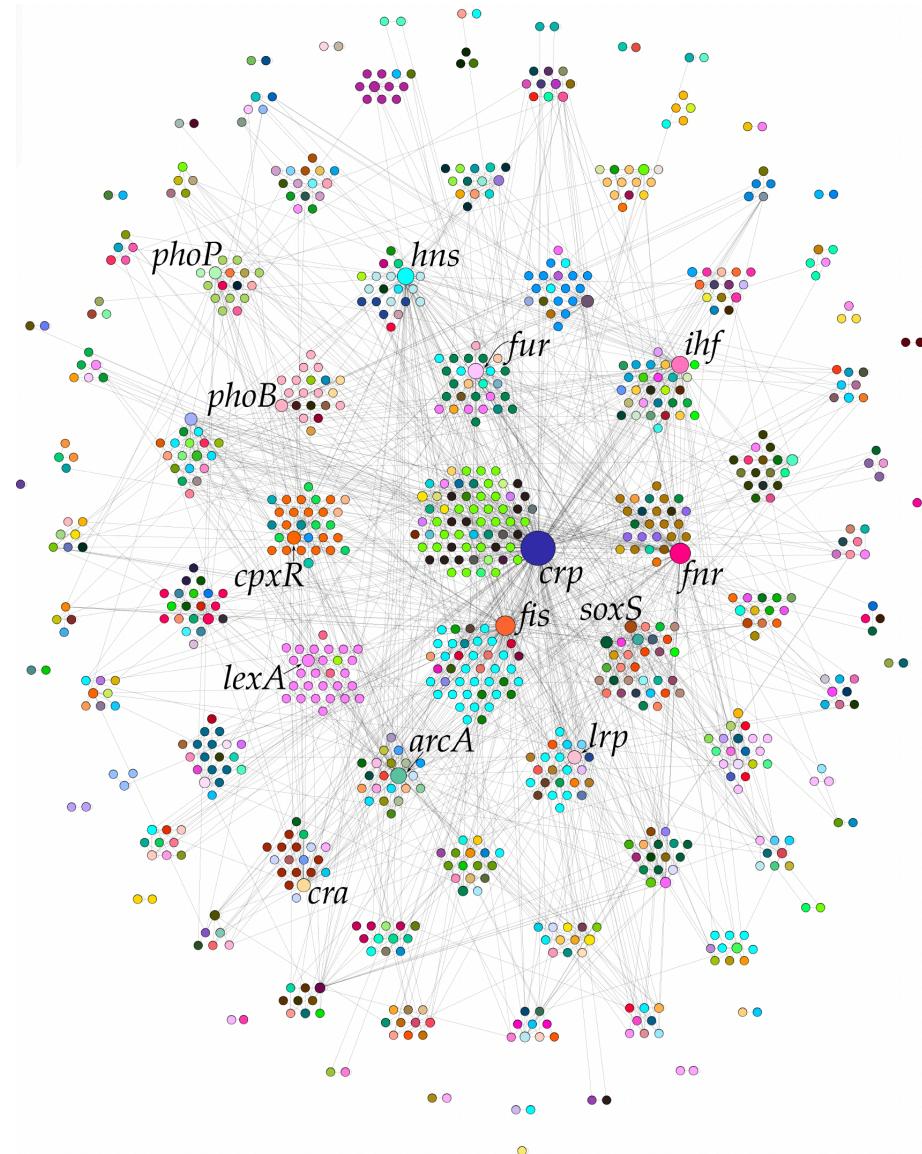
Biological networks are modular (Hartwell et al., 1999a; Brugere et al., 2018; Pratapa et al., 2020; Marbach et al., 2012; Girvan and Newman, 2002) and modularity and community detection algorithms have been used to identify functional modules in biological networks (Girvan and Newman, 2002). Given the different ways to partition a network, it is of interest to compare them not only at the topological level but, most importantly, on their functional significance in the cell. Below, we examine the different partitions in TRNs and metabolic networks.

17.5.1 Fibers and modules in TRNs

We ran the Louvain community detection algorithm on the *E. coli* TRN studied in Chapter 14 and found the modular partition of Fig. 17.7. We used the resolution parameter 0.16 in the Louvain algorithm (Blondel et al., 2008), which reaches the highest modularity (the density of links inside communities as compared to links between communities) at $Q = 0.516$. Other parameters yield similar results. A visual comparison indicates that the modules identified by the Louvain algorithm do not capture the partition provided by the fibers. This is corroborated when we calculate the performance of the Louvain modularity algorithm as the fraction of genes classified correctly in the fiber, which is just 23%. The results indicate that in the TRN, the functional partition of the genetic network into synchronized fibers cannot be captured by the modularity detection algorithm.

The functional modules of synchronized genes that we identify using fibrations in the TRN are difficult to detect with the community detection algorithm. This is related to the role of hubs. A modularity algorithm will put in the same module a hub like *crp* in the carbon SCC shown in Fig. 14.3 together with all its regulated genes since they are all connected. But dynamically this is not the case since *crp* does not synchronize with its regulated genes (they belong to different fibers). Fibrations, on the other hand, correctly classify the hub *crp* as an independent regulator not synchronized with any gene in its regulon. This is not an isolated example. Figure 17.7 shows that the Louvain modularity algorithm captures only 23% of the synchronized functional fibers.

In general, modularity analysis may not capture functional modularity when applied to a structural biological network like a TRN, although in the next section, we see that the

**Fig. 17.7**

Modularity versus fibers in TRN. Louvain modularity detection algorithm applied to the TRN of *E. coli*. The modules obtained are clustered in space and the color of the genes indicates the fibers. Typically, modules are composed of many fibers, indicating that the fiber partition of synchronized genes is different than modularity.

fibers may be included within a module. However, modularity detection is applicable to a biological functional network, as obtained by thresholding a correlation matrix of fMRI BOLD signals in the brain or coexpression. The difference between a structural network and a functional network is investigated in Chapter 20. In functional networks, similarity

in coexpression indicates synchronization and, therefore, functional significance. Thus, a modularity algorithm works well to find functional modules in functional networks but not in a structural network like the TRN or a connectome (see Section 23.5.4 for an application in the brain). In short, we have seen plenty of examples where genes are not connected in the TRN but are still synchronized, as in a multi-layer fiber. In these cases, fibration analysis should be used to identify the fibers associated with function in the structural network. We develop this idea further in Chapter 20.

17.5.2 Fibers, modules and motifs in metabolic networks

The overwhelming presence of fibration building blocks and their intricate cycle structure in the enzyme networks studied in Chapter 16 prompted Álvarez-García et al. (2025b) to compare their characteristics and biological significance to network motifs and modules (Fig. 17.8).

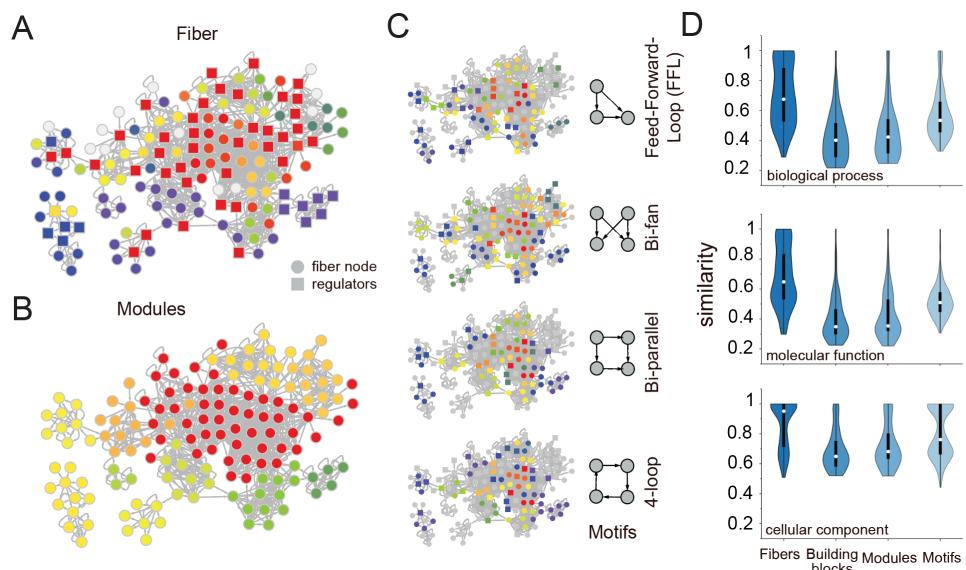


Fig. 17.8

Biological significance of fibers, modules and motifs. (a) Fibers in the carbon enzyme network, where colors refer to proteins in the same fiber. (b) Topological clusters were found using the Louvain algorithm. (c) Presence of significant motifs of different types. (d) Violin plots of the average functional similarity of enzyme pairs in fibers, clusters, motifs, and fibration building blocks using GO terms from the molecular function (MF), biological processes (BP), and cellular components (CC) ontologies. Enzymes in fibers show significantly higher similarities compared to enzymes in cluster and motif building blocks ($P < 10^{-3}$, Mann-Whitney U-test). Figure reproduced from (Álvarez-García et al., 2025b).

As a consequence of the fundamentally different ways of detection, we observe that

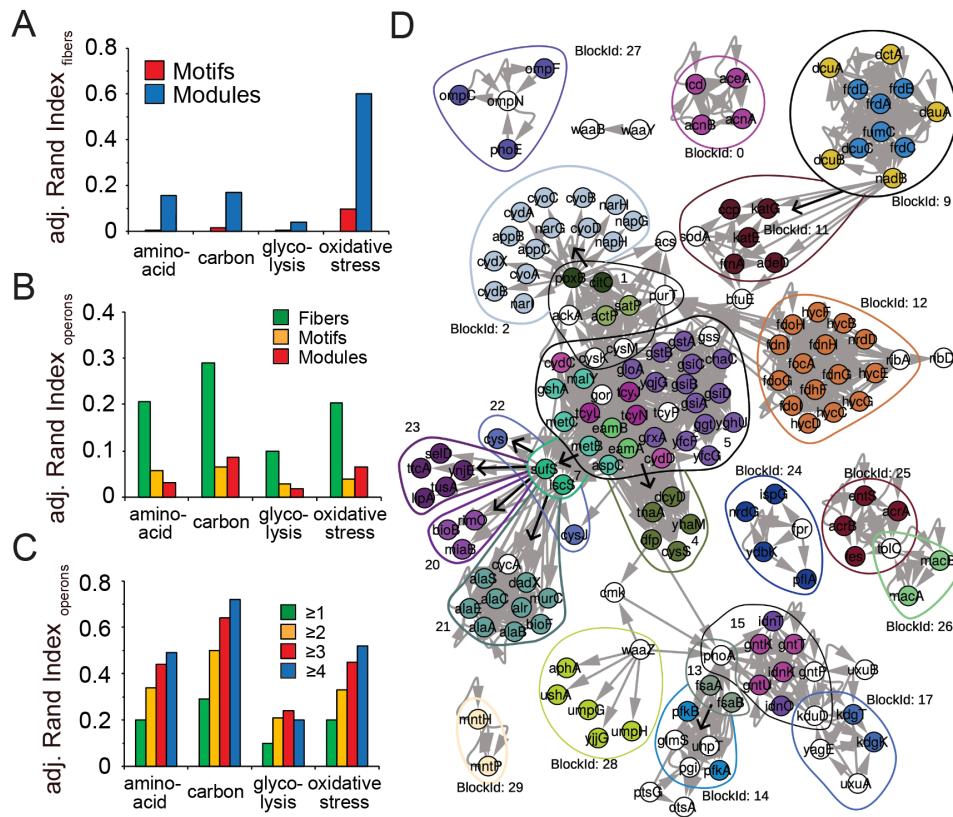
motifs and modules are significantly less functionally similar than fibers, suggesting that coherence between genes is a better indicator of biological significance than the over-representation of topological patterns. Furthermore, motifs fail to complement fibers, while network modules show the opposite. Such finding is probably rooted in the observation that modules partition nodes in the network, increasing the chances that such groups of nodes harbor fibers as well. Furthermore, the underlying fibers are bundles of enzymes where metabolic information flows coherently, suggesting that such patterns may also be reflected in the genomic arrangements of genes in operons and fibers in the transcriptional regulatory network. Indeed, we find that the overlap between fibers is better established than motifs or modules, suggesting that fibers point to biologically relevant, elementary building blocks.

Focusing on the carbon metabolism subnetwork of enzymes, we determine modules using the Louvain algorithm and search for the most popular network motifs: feed-forward-loops, Bi-Fan, Bi-parallel and 4-cycle motifs (Shen-Orr et al., 2002; Milo et al., 2002). A visual inspection of the coverage of fibers, modules, and motifs in Fig. 17.8a-c indicates that modules partition nodes in the underlying networks, while fibers and motifs, as expected, cover the underlying network only partially. In particular, the determination of modules and motifs is based on the connectivity of the underlying network, while fibers account for similarities of input trees. As a consequence, motifs and modules merely reflect local connectivity. In contrast, fibers bundle together genes that share the same information flow, suggesting that synchrony transcends simple connectivity. In other words, nodes can be synchronized (*i.e.* share the same input tree) without being necessarily connected in contrast with motifs and modules.

Given that fibers bundle information flow, as exemplified by their input trees, we hypothesize that the functional roles of enzymes in the same fiber should show a higher degree of functional similarity than motifs and modules that are based on simple connectivity. Furthermore, motifs are based only on statistical significance, which may not be enough to guarantee homogeneous functionality.

As a benchmark, we compare the functional similarity of fibers to motifs and modules, given the previous observations that strongly interacting proteins have a heightened propensity to be involved in similar biological functions and to appear in the same cellular components. In Fig. 17.8c, we observe that the average functional similarity of enzyme pairs in fibers is higher than their module and motif counterparts in all enzyme networks using GO terms from all three ontologies. Similar results are found in all metabolic networks separately studied in (Álvarez-García et al., 2025b). We further consider the functional similarity of building blocks that, in the most complex ways are composed of many fibers, such as composite feed-forward and feedback Fibonaccis. As for their functional similarity, we still find that building blocks have similarities that compare to modules (Fig. 17.8d).

Furthermore, we wonder to what extent fibers, modules and motifs complement each other. Determining an adjusted Rand Index, we find that modules overlap significantly better with fibers than motifs in all four metabolic networks (Fig. 17.9a). Such an observation is rooted in the fact that modules arise from partitioning all nodes in the underlying network, increasing the chance that a large module harbors comparatively small fibers. Very few nodes participate in more than one motif or fiber, which reduces the chance that such structures substantially overlap.

**Fig. 17.9**

Comparison between fibers, modules, and motifs in the enzyme network. (a) Adjusted Ward index as a measure of the similarity of two partitions. Fibers overlap better with modules than with motifs in all enzyme networks. (b) Similarity between genes in operons and fibers, modules, and motifs indicate that fiber gene sets show the greatest overlap with operon gene sets. (c) The propensity of fibers in the underlying metabolic networks to overlap with genes in operons increases with elevated operon size. (d) Map of building blocks (enclosed by closed lines) and their constitutive fibers (nodes in color) in the oxidative stress network. Some regulators may belong to more than one building block, while the fibers are disjoint. Figure reproduced from (Álvarez-García et al., 2025b).

As another indication of the biological relevance of genes that are organized in fibers, we determine the overlap with genes that appear in operons in *E. coli*, assuming that symmetries that dominate the network topology are reflected in the regulatory organization of gene sets. Utilizing 2,579 operon gene sets as of the RegulonDB databases, we determine overlaps using an adjusted Rand Index, and find that fibers overlap significantly better with operons than motifs and modules in all four different metabolic networks (Fig. 17.9b). Furthermore,

we refine our analysis by considering operons that consist of at least a certain number of genes. Fibers increasingly overlap with operons that harbor more genes (Fig. 17.9c).

These results suggest that fibers indeed bundle and potentially coordinate enzymes that are involved in the same functions, pointing to elementary building blocks in the underlying networks. In particular, such fibers do not necessarily overlap with canonical metabolic pathways, but offer a novel perspective for finding new pathways.

Figure 17.9d maps the fibers and building blocks in the oxidative stress network. As an example, we observe a fiber that is composed of the *fduABCD* operon of genes building the fumarate reductase enzyme complex (BlockId 9, Fig. 17.9d). Such a fiber is entangled with another fiber in a complex feedback loop that is composed of subunits of the anaerobic C-4-dicarboxylate transporter *dcuABC* and *dctA*, which is responsible for the uptake of fumarate, succinate, L-aspartate and L- and D-malate under aerobic conditions. While the underlying connections between the single enzymes that are involved in the transport of metabolites and corresponding reductase is highly complex, we break the underlying complexity down to two fibers that not only bundle the underlying functions effectively but also indicate which functions are connected to each other on a dynamic basis. In other words, the complex multiplicity links between enzymes are mapped to relatively simple building blocks of enzymes that are dependent on each other, pointing to the relevance of fibers as representatives of potential pathways.

The present evidence indicates that enzymes in fibers are functionally more homogeneous than enzymes in network motifs or network modules. This observation points to a hitherto unknown level of complexity in the organization and architecture of biological networks, which topological motifs and modules through statistical means simply miss. As a consequence, such fibers may harbor more functional information compared to motifs and modules.

Additionally, fibers might offer a superior approach to uncovering pathways from a different perspective compared to motifs and modules. They could also help identify novel pathways that are obscured within existing representations of biological information. Furthermore, fibers could be utilized to discover new drug targets, providing an innovative means to pinpoint potential therapeutic interventions.

We have seen how the functions of biological networks can be pictured as an orchestra of synchronized instruments captured by the fibration symmetry. In this chapter we elaborate on an even stricter and more taxing criterion for functionality of the network. We ask whether minimal biological circuits can perform core logic computational programs. We show this through the mechanism of ‘fibration symmetry breaking’. These circuits can then be used to build computational molecular machineries. This helps in system biology to design biological machines from the bottom up, following a systematic approach from first principles of symmetry and symmetry breaking.

18.1 Synthetic biology design through symmetry and broken symmetry

Synthetic biology attempts to build circuits and systems to implement specific functions. A long tradition at the interface of biology and engineering proposes that the functional building blocks of these systems should offer computational repertoires drawing parallels between biological networks and electronic circuits. Indeed, the idea of using electronic circuitry and devices to mimic aspects of gene regulatory networks has been in circulation since the inception of regulatory genetics by Jacob and Monod (1961); Jacob (1977). It has been a driving force in synthetic biology, with several demonstrations showing that engineered biological circuits can perform computations analogous to electronic circuits and computers.

In 2000, the first two synthetic circuits were built in bacteria. These circuits mimic two core functionalities of computer systems: timekeeping clocks (oscillators), and memory. A major breakthrough was the demonstration of a synthetic genetic oscillator by Elowitz and Leibler (2000): the repressilator composed of three genes in a ring interacting by repressors, discussed in Section 8.9.2.

Another basic circuit with oscillatory behavior is a purely negative feedback loop, comprising two genes A and B , where A inhibits B and B activates A . This circuit by itself could function as a pulse generator since it is capable only of damped oscillations (Alon, 2019). However, in the presence of noise it generates sustained oscillations with a reliable frequency but ill-defined amplitude. Some simple additions can be made to this basic structure to improve its oscillatory behavior. For example, Purcell et al. (2010) show that from an analytical standpoint the addition of an activation self-loop in gene B provides an amplified negative feedback loop; however, only damped oscillations have been seen *in vivo* (Purcell

et al., 2010). Furthermore, addition of an inhibiting self-loop in gene *A* gives the Smolen oscillator of Section 8.8.3, which exhibits sustained oscillations (Stricker et al., 2008) in a robust and highly tunable manner.

A second major advance was made by Gardner et al. (2000), who demonstrated a genetic toggle-switch allowing memory storage in the form of a bit (discussed in Section 8.9). The toggle-switch is a bistable two-way switch, analogous to an electronic flip-flop that functions as a binary memory in a computer. Bistability is the key feature of this circuit: it can be switched between two stable equilibria by different inputs. Gardner et al. (2000) showed the property of bistability in this circuit using the concentrations of two repressors and their effective synthesis rates.

Another prominent design to store memory is the lock-on circuit, discussed in Section 8.8.1, which, unlike the toggle-switch, stores memory in a ‘permanent’ form. This is a genetic circuit consisting of a positive autoregulation (PAR) feedback loop, which works as a bistable one-way switch circuit sustaining two stable states: both genes inactive or both genes active after either one has been activated (Tyson et al., 2003). In contrast to the toggle-switch, the lock-on is a one-way switch: once the circuit switches to the activated case it cannot return to its previous state. This type of lock-on circuit take a role in developmental processes characterized by a state transition, such as apoptosis (programmed cellular death) (Tyson et al., 2003).

Since the introduction of the repressilator and toggle-switch two decades ago, an explosion of activity (Cameron et al., 2014; Khalil and Collins, 2010) has demonstrated a myriad of genetic circuits that are able to perform basic logical operations necessary for a computational device (Dalchau et al., 2018; Tanenbaum, 2016; Tyson et al., 2003; Khalil and Collins, 2010; Fussenegger, 2010). Such circuits are constructed using feedback loops, both positive and negative, (Tyson et al., 2003; Dalchau et al., 2018) and are executed by synthetic switches and oscillators designed from simple components such as interacting genes or protein-protein interactions. They includes various toggle-switches for memory storage (Gardner et al., 2000; Atkinson et al., 2003; Kramer et al., 2004; Kramer and Fussenegger, 2005; Ajo-Franklin et al., 2007; Ham et al., 2008) and logic operations (Anderson et al., 2007; Guet et al., 2002; Rackham and Chin, 2005), oscillators (Elowitz and Leibler, 2000; Atkinson et al., 2003; Fung et al., 2005; Stricker et al., 2008; Tigges et al., 2009), pulse generators (Basu et al., 2004), and edge detectors (Tabor et al., 2009).

Remarkably, Leifer et al. (2020) and Álvarez-García et al. (2025a) find the presence of circuits closely resembling of all these synthetic circuits at the core of the *E. coli* TRN. This suggests a view of the minimal TRN core as a logical computational machine (Kondev, 2014), to be discussed in Chapter 19. In this chapter we perform a systematic analysis of circuit capable of logical computations in TRNs and present algorithms to seek them in any network through fibration symmetry breaking. This chapter thus extends Chapter 15 by considering electronic analogs.

Since these circuits can be constructed artificially to perform computations, it is reasonable to expect to observe them, or some close variation, in the core computational subset of the network. We expect to find memory storage circuits, as well as oscillating circuits for timekeeping. In the case of the TRN of simple model bacteria like *E. coli*, we expect to observe the simpler forms of these known genetic circuits from synthetic biology, which

is indeed the case, as will be shown next. These circuits can be understood through the concept of a broken-symmetry fibration, introduced by Leifer et al. (2020).

We formalize this quest mathematically, by showing that this principle of symmetry breaking, when applied to circuits with symmetries, reveals a hierarchy of genetic circuits across species from bacteria to humans. These circuits map to the fundamental building blocks of electronic architectures (Horowitz and Hill, 2015): starting with the transistor and progressing to mirror circuits, ring oscillators, and complex logic integrated circuits involving flip-flops. The functionality of these circuits is analogous to electronic operations: they act like clocks and counters in their symmetric states, and as toggles, latches and memory storage units (flip-flops) in their symmetry breaking states.

So far we have seen that a large part of the regulatory network is made of synchronized fibers. In the case of *E. coli*, 66% of the genes can be characterized by symmetries. In this chapter we show that the remaining genes can also be accounted for by symmetry breaking (and a few more regulatory functions), so we can classify *every single gene* in the bacterium's genome by its function. Beyond the fibers identified by surjective symmetry fibrations, the network can be further reduced by the application of an injective fibration, which identifies a reduced driver networks elaborated in Chapter 19. The remaining network is organized into three giant strongly connected components, which control the fibers. We further decompose these components into circuits identified by broken fibration symmetries. The circuits play a crucial functional role in the network, acting as memory storage flip-flops analogous to building blocks of memory in digital computers. These logical genetic circuits are mainly composed of feedback loops (Álvarez-García et al., 2025a) or frustrated topologies (Leifer et al., 2020). Negative feedback loops (or *frustrated* circuits) give rise to oscillating circuits, while positive feedback loops allow for memory storage.

18.2 A biological transistor at the core of genetic circuits

To understand the computational rationale for symmetric and symmetry-broken circuits made of repressors, we map them to electronic analogs. We start our analogy with electronic circuits with the simple fibration building blocks found in the *E. coli* TRN displayed in Fig. 15.2, spanning from the AR loop to the FFF.

A first realization is that a repressor link between a source and target gene in a TRN and can be mapped to a NOT gate, which is a primary function of the transistor (Fig. 18.1a). A repressor or inhibitory interaction is, in turn, the primary interaction of all biological systems, from genetic networks to the brain.

The mappings between electronic and gene regulatory circuits are straightforward. Even though nothing is circulating in a genetic circuit, the information flow between genes represented by the TF can be thought of as circulating charges in electrical circuits. The expression level of a gene is analogous to the voltage in an electrical circuit. A battery does work to produce a given voltage level; likewise a gene does work to produce a desired expression level. Connecting a battery to a device through a wire is like connecting a gene

to another one through a regulatory link; the electric current represents the rate of change of the expression levels in the genetic circuit.

18.2.1 Repressor interaction maps to a transistor

We begin the analogy with the simplest circuit: a single gene with a feedback loop with repression (AR loop, Fig. 18.1d). At the simplest Boolean level, the dynamics for the expression level y_t is described by the discrete time model with Boolean interaction (Glass and Kauffman, 1973):

$$\Delta y_t = y_{t+1} - y_t = -\alpha y_t + \gamma_y \theta(k_y - y_t). \quad (18.1)$$

Here, α is the degradation rate of the Y gene product, γ_y is the maximum expression rate of gene Y, and k_y is the dissociation constant. The Heaviside step function $\theta(k_y - y_t)$ reflects the repressor autoregulation in the Boolean logic approximation.

Leifer et al. (2020) show that the genetic repressor interaction, shown as the stub in Fig. 18.1b, is the genetic analog of the solid-state transistor, shown in Fig. 18.1a.

The formal (mathematical) analogy mediated by the charge transport process in semiconductor materials indicates that the repressor interaction, symbolized by the $\theta(k_y - y)$ term in (18.1), works like a logic NOT gate. At the foundations of electronics, it was understood that a NOT gate has to convert a low input into a high output, and thus it requires an amplifier, which cannot be made out of a combination of diodes. The device that does the job is the transistor.

A transistor is typically made up of three semiconductors, a base sandwiched between an emitter and a collector (Fig. 18.1a). The current flows between the emitter and collector only if voltage applied to the base is lower than at the emitter ($V_B < V_E$) and thus the transistor acts as a switch and inverter. In the genetic circuit, the expression y_t drives the rate of expression of gene Y, like the voltage drives current around an electric circuit. Comparing (18.1) to the pnp transistor in Fig. 18.1a leads to an analogy in which the expression y_t is an analog for the base potential V_B of a transistor, k_y is an analog for V_E , γ_y is an analog for the emitter current I_E , αy_t is an analog for I_B , and Δy_t is an analog for I_C . Then (18.1) provides the genetic equivalent of the equation for a transistor's collector current $I_C = I_E - I_B$ (Fig. 18.1cd).

The mapping between a transistor and a repressor is formalized in (Fig. 18.1a, b):

$$V_E = k_y \quad V_B = y_t \quad V_C = y_{t+1} \quad I_E = \gamma_y \quad I_B = \alpha y_t \quad I_C = y_{t+1} - y_t. \quad (18.2)$$

18.2.2 AR loop maps to a ring oscillator

In the AR loop shown in Fig. 18.1d the repressor link connects to its own gene. Analogously, the collector of the transistor connects to the base. Thus, the repressor AR genetic circuit of Fig. 18.1d becomes a one-stage ring oscillator (Fig. 18.1c) where the collector of the transistor connects to the base forming the minimal signal feedback loop. As shown in Fig. 18.1d, an example of the AR is the gene Y = *trpR* from the *E. coli* transcriptional network.

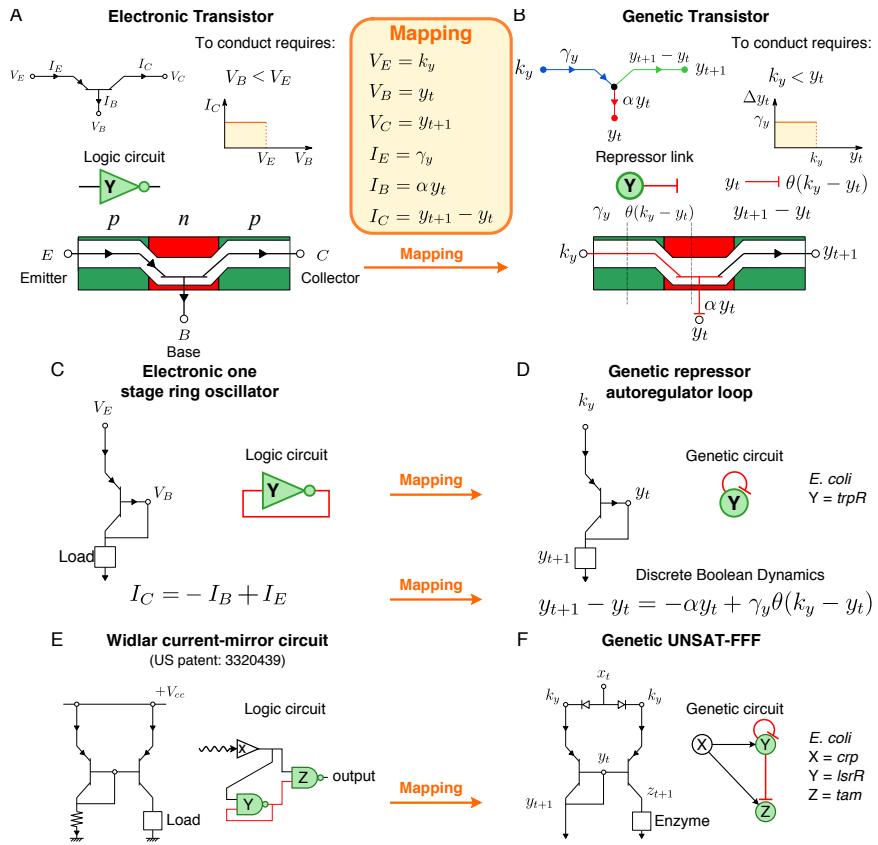


Fig. 18.1

Mapping between electronic and biological symmetry fibration circuits. (a) A pnp transistor allows current flow if the voltage applied to its base is lower than the voltage at its emitter ($V_B < V_E$). Since it has a high (low) output for a lower (high) input, it is logically represented by a NOT gate. The yellow box shows the mapping between the pnp transistor and the biological repressor. (b) A repressor regulation link plays the role of the pnp transistor since the rate of expression of a gene is repressed by gene Y if $k_y < y_t$. (c) Connecting the base of the transistor to its collector forms a one stage ring oscillator. (d) This connection is translated to the biological analog as a repressor autoregulation at gene Y. In this way, the rate expression of gene Y is able to oscillate, depending on the parameters α , k_y and γ_y . (e) Widlar current-mirror circuit and (f) its biological analog (UNSAT-FFF). By mirroring the ring oscillator, the Widlar mirror circuit allows synchronization and oscillations. (g) Phase diagram of oscillations of the UNSAT-FFF. An oscillatory phase is defined by the condition $\gamma_y/k_y > (\gamma_x/\alpha)^{-1}$. For instance, on the right side we plot the solution of the discrete dynamics for a set of parameters satisfying such a condition. Specifically, $\alpha = 0.205$, $\gamma_x = 0.454$, $\gamma_y = 0.454$, $k_x = 0.5$, and $k_y = 1.0$. Figure reproduced from (Leifer et al., 2020). Copyright © 2020, Leifer et al.

18.2.3 FFF maps to a Widlar current-mirror electronic circuit

The repressor AR loop can be extended to the FFF by symmetrizing it, adding gene Z, such that it synchronizes with Y to express an enzyme that catalyzes a biochemical reaction (Fig. 18.1f). The circuit is completed by an external regulator X that maintains the symmetry between Y and Z. The resulting UNSAT-FFF is shown in Fig. 18.1f as *E. coli*'s X = *crp*, Y = *lsrR*, Z = *tam*. Leifer et al. (2020) show that the frustrated topology UNSAT-FFF $|n = 1, l = 1\rangle$ give rise to oscillatory behavior (Section 15.3.4).

The UNSAT-FFF maps to the so-called Widlar current-mirror electronic circuit shown in Fig. 18.1e, a popular building block of integrated circuits used since the foundation of the semiconductor industry—1967 US patent (Widlar, 1969b, 1967, 1969a; Horowitz and Hill, 2015). It serves two key functions as we show below: mirror synchronization of $y_t = z_t$ and oscillatory activity (Fig. 18.1g) which could be used as a timekeeping clock.

Interestingly, a single self-inhibiting gene can also exhibit oscillatory behavior if the self-inhibition signal exhibits a delay as shown in Section 15.3.4, allowing for its expression levels to increase before its self-inhibition decreases them. This is actually a very sensible prediction, given that there is some natural delay in the system while the gene is transcribed, then translated into a TF and finally binds to the gene's promoter region, which requires the concentration of the TF to be high enough for binding to take place.

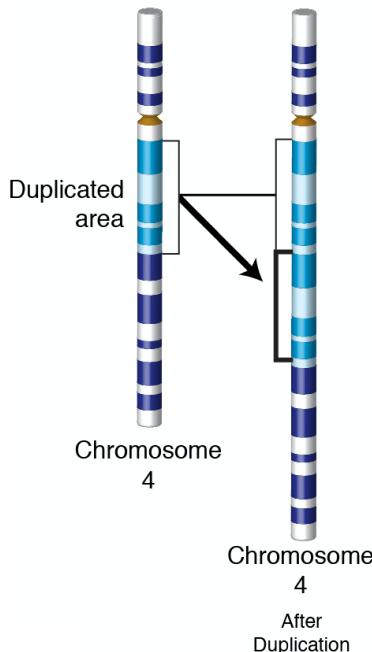
18.3 Gene duplication in evolution mimics lifting in a fibration

Chapter 11 discussed the robustness of a genetic network evolving by gene duplication which generates rich fibrations as compared to one with only automorphisms. Figure 11.1 exemplifies, anecdotally, how gene duplication and divergence can lead to a robust fiber-rich gene network.

Mathematically, fibrations are built by the lifting property, and this mechanism can be mapped to the duplication of genes and edges in the biological graph. This suggests that the emergence of fibration symmetry in the TRN is a major design principle of genetic networks, consistent with their dynamical evolution as fibrations are created. Gene duplication (see Fig. 18.2) is a crucial mechanism in the evolution of genomes and genetic systems. This process generates new genes by duplicating existing ones when a segment of the chromosome that contains a gene is replicated, resulting in the cell possessing two paralogous copies of a set of genes.

Gene duplications typically occur due to errors in DNA replication and repair mechanisms. This process not only duplicates the gene itself but also its promoter region—the DNA sequence adjacent to the gene that contains the binding sites for transcription factors (TFs). This duplication creates redundancy within the cell.

Below we show that gene duplication is analogous to the lifting operation that defines the fibration (see Section 6.5). This strongly suggest an evolutionary drive towards fibrations symmetries in the genome. It also provides a natural model for the evolution of fibration

**Fig. 18.2**

Duplication event in the genome. A region of the chromosome is duplicated copying a number of genes. When the duplication follows the lifting property, the event increases the number of genes in a fiber without affecting their synchronization. Thus, it is analogous to a gauge transformation in the genome as discussed in Section 12.6.2. Reprinted from Wikipedia https://en.wikipedia.org/wiki/Gene_duplication.

symmetry and broken symmetry circuits in genetic networks. It defines a hierarchy of broken symmetry circuits with toggle-switch memory functionalities like the electronic flip-flop. At the same time, the fibration building blocks of the previous section work as clocks with varying levels of sophistication. Symmetry and broken symmetry circuits provide computing functionalities to the cell composing all building blocks of TRNs.

The evolutionary design principle of the TRN follows the lifting of genes and edges from a primordial set of fibration bases depicted in Fig. 18.3. At the most basic level this leads to the simple building blocks described in Chapter 15. Starting from the base of these building blocks, by duplication we obtain symmetric fibers, and then by breaking their symmetries we obtain logic circuits for memory storage and timekeeping. This lifting-driven dynamical evolution creates the core of the computational decision-making of the cell, which constitutes the apparatus of its genetic collective intelligent device.

We explain this new hierarchy next.

Gene duplication has been proposed as the mechanism behind the emergence of motifs (Milo et al., 2002; Shen-Orr et al., 2002; Mangan and Alon, 2003). Fibers could emerge in an evolutionary setting of random rewiring (mutation) and selection for functional structures.

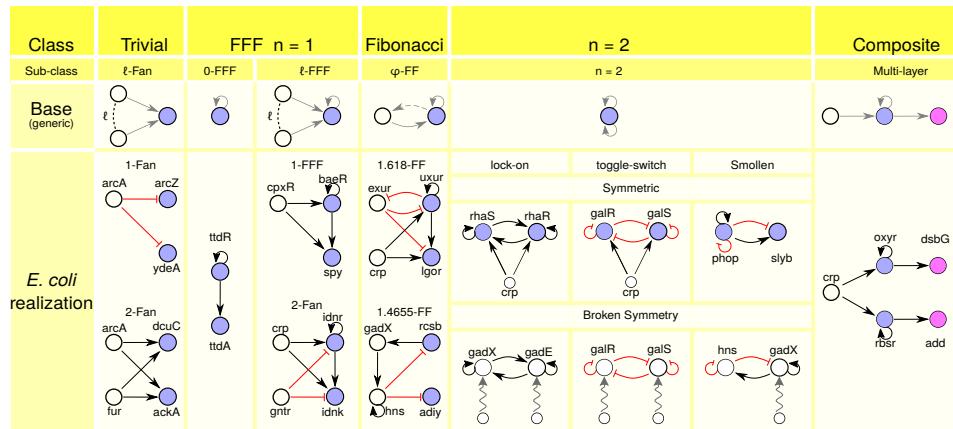


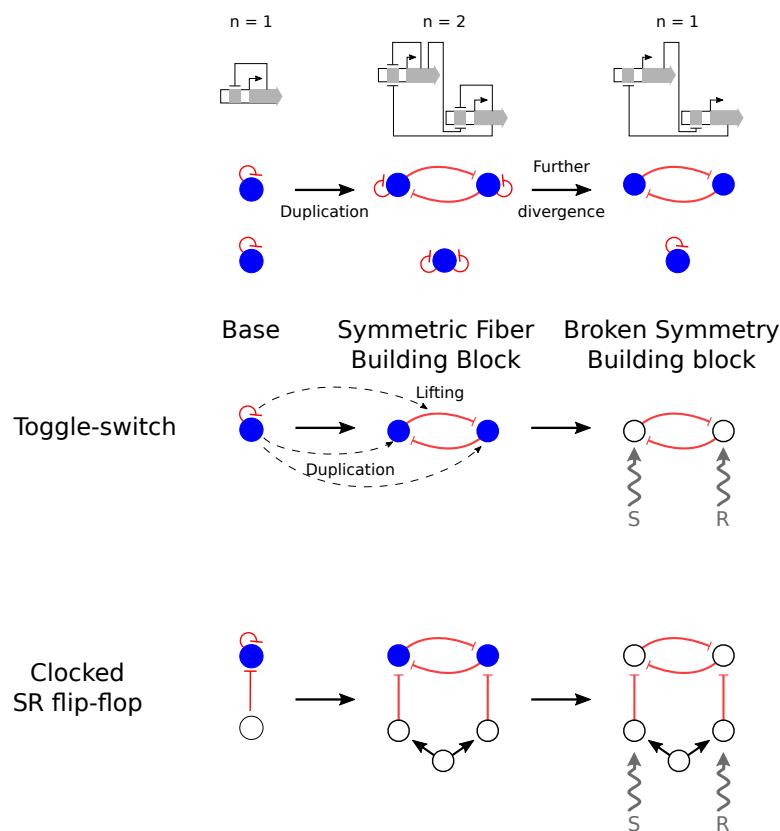
Fig. 18.3

Bases of simple fibration building blocks in TRNs. These circuits serve as bases for duplicated circuits leading to a broken symmetry circuit performing as a flip-flop. TRNs can be seen as assemblies of five basic fibration building blocks (top row): (i) Trivial fibers with $|n = 0, l \neq 0\rangle$. (ii) The AR loop $|n = 1, l = 0\rangle$ and FFF $|n = 1, l \neq 0\rangle$. (iii) The Fibonacci fiber, φ -FF. (iv) the $|n = 2, l\rangle$ fibers. When this symmetry is broken it forms the memory and oscillatory circuits. (v) Multilayer fibers of the previous ones. By adding different types of the previous four building blocks in a sequential manner, a composite fiber is obtained. Figure reproduced from (Álvarez-García et al., 2025a).

However, such evolution would probably be slow. In contrast, gene duplication generates fiber structures fast and ‘for free’, with subsequent minor modifications of the duplicated arrows (mutations in the regulatory region modifying incoming edges, or in the coding region modifying outgoing edges).

In evolution, multiple parologue copies of a gene may emerge via gene duplication events, and subsequent evolution may lead to differences in their coding or regulatory regions. If the gene codes for a transcription factor, the duplication will correspond to a duplication of a node in the TRN, and the subsequent changes may cause a subsequent rewiring of outgoing or incoming edges, respectively. The first step, the process of taking a gene and ‘opening it up’ into one or multiple other identical genes, is akin to the lifting operation between a collapsed base graph and its original graph. The lifting operation can be understood as opening up a collapsed fiber node in the base into the full set of original synchronous nodes that share the same color. The key feature of the lifting operation is that the nodes obtained by this opening up have the exact same inputs and input trees as the node that is being opened up. This ensures that the new network’s dynamics is the same as the original collapsed base graph. This is a great property to have if the duplication event is really duplicating the dynamics as well. That is, this is a change without a change to the cell; a symmetry.

The cell is then free to diverge. Further mutations in the coding region then cause the two original copies of the gene to perform different functions, and thus diversify the bacteria’s

**Fig. 18.4**

Broken symmetry circuits and duplication/lifting process. How flip-flops emerged by the lifting operation analogous to duplication and broken symmetries. Starting with an initial simple structure a gene duplication process can create a symmetric fiber structure, analogous to a lifting process. The structure resulting from the duplication may undergo further modifications (top). The circuit formed depends on the initial structure that is replicated (bottom). For each circuit, the symmetric form is at the center, while the broken symmetry process is on the right. The *set-reset* (*S-R*) inputs break the symmetry by sending different regulations to an otherwise synchronous pair of nodes. The *S-R* inputs can be any different nodes sending different signals or can be the same node sending different signals to the nodes in the fiber. Figure reproduced from (Álvarez-García et al., 2025a).

functions while creating bigger fibers. This process has been described also in Section 12.6.2 in terms of curvature and connection in the genetic network.

This process does not only duplicate existing genes. The most important part is the subsequent divergence. By virtue of creating a duplicate of the previous genes with identical dynamics, one of the genes is ready to mutate and diverge. In particular, the symmetry now can be broken. If the symmetry of some of the symmetric building blocks obtained by

duplication is broken, logical circuits are obtained as we explain next. By taking the basic fiber building blocks in Fig. 18.3 and duplicating their bases we obtain different replica circuits still preserving the same dynamics and synchrony. Nevertheless, if there are added regulations that break symmetry, the result is a circuit that performs logical functions: a symmetry-broken circuit.

It is important to point out, however, that there exist duplication processes that do not preserve the inputs. For example, a gene with self-regulation, once duplicated, would still have self-regulation, but would also have cross-regulation with its copy, meaning it has gone from having one input to two, which changes the original input tree; see Fig. 18.4. Therefore, evolution or some other mechanism, is needed to ensure at least some duplication events preserve the input tree of the original gene. This implies that evolution is selecting for duplications that preserve the dynamics of the network, like lifting-driven duplication.

18.4 Symmetry breaking in physics

While symmetry principles stand as crucial elements within natural laws, much of the world's complexity emerges from mechanisms of symmetry breaking, which encompasses various ways in which nature's symmetry can be veiled or disrupted (Anderson, 1972). Any situation in physics, in which the ground state (i.e., the state of minimum energy) of a system has less symmetry than the system itself, exhibits the phenomenon of symmetry breaking. For instance, the phases of matter are characterized by different symmetries. At higher temperatures, matter takes on a ‘higher symmetry’ phase (e.g., liquid, paramagnetism, normal conductivity, and fluidity), while at lower temperatures, the symmetries of the phases are broken to ‘lower symmetry’ (e.g., solid, ferromagnetism, superconductivity, and superfluidity).

Most symmetry laws in physics are broken in one way or another. One such mechanism is spontaneous symmetry breaking, where the laws of physics remain symmetric, but the system's ground state exhibits a lower symmetry than the full system, as in a paramagnetic-to-ferromagnetic phase transition. For temperatures below the critical value T_c , the magnetic moments of the atoms of a ferromagnetic material are partially aligned within magnetic domains, producing a net magnetic moment even though the atoms interact through a spin-spin interaction, which is invariant under rotation. Thus the rotationally invariant symmetry of the system is broken to this ground state, which has nonzero magnetic moment. As the temperature increases, this alignment is destroyed by thermal fluctuations and the net magnetization progressively reduces until it vanishes at T_c . The orientation of the magnetization is then random. Each possible direction is equally likely to occur, but only one is chosen at random, resulting in a zero net magnetic moment. So the rotational symmetry of the ferromagnet is manifest for $T > T_c$ with zero magnetic moment, but is broken by the arbitrary selection of a particular (less symmetric) ground state with non-zero magnetic moment for $T < T_c$.

Another type is explicit symmetry breaking, where the dynamics is only approximately symmetric, yet the deviation caused by the breaking forces is minimal. Now we can consider

the symmetry violation as a small correction to the system. An example is the spectral line splitting in the Zeeman effect due to a magnetic interaction perturbing the Hamiltonian of the atoms involved.

The type of symmetry breaking we find in genetic circuits is more of an explicit type. The lifting of the base produces a symmetry that is explicitly broken by external regulators, for instance S and R in a flip-flop.

18.5 Hierarchy of broken symmetry circuits: duplication and memory

Álvarez-García et al. (2025a) and Leifer et al. (2020) find that circuits with broken fibration symmetry act as circuits performing basic logic computations of memory storage in the TRN like toggle-switches and flip-flops in electronics (Fig. 18.4). These circuits are identified by starting with a symmetric circuit, which originates from a fiber building block, and breaking its symmetry by adding extra edges acting as regulators that break the symmetry in the fiber structure. The breaking of symmetry is the result of a process of gene duplication that starts with the symmetric base of a fiber building block, which is duplicated to a (still) symmetric structure. This new structure may then incorporate new regulations that result in symmetry breaking of this duplicated circuit, as seen in Fig. 18.4. Crucially, if the result of a duplication process respects the lifting property and preserves the fibration symmetry (Fig. 18.3), this new symmetric structure is prone to obtain new regulations that result in symmetry breaking.

Strictly speaking, the duplication process of a base does not necessarily produce a symmetric structure or a structure with the same fiber classification. For example, if an AR repressor gene is duplicated into two genes (Fig. 18.4, top), a precise duplication of the auto-repressor should produce a structure in which the two parologue genes mutually repress each other as well as also auto-repress themselves. The original structure corresponds to a $|n = 1, l = 0\rangle$ AR loop, while the new duplicated structure would in fact correspond to a $|n = 2, l = 0\rangle$ binary-tree instead, which accounts for the two interactions that the new parologue genes receive.

However, the initially duplicated structure may undergo further divergent modifications, like the removal of the self-inhibitions in this case, which result in a structure that preserves the same fiber classification since the input relations are the same as in the original structure. This means that the duplicated circuit can be lifted from its AR base, or, analogously, application of the fibration to the duplicated circuit results in the base.

This leads to a structure with the same dynamics as the base, by respecting the lifting process. This new structure, when collapsed via a symmetry fibration, gives the original structure that was replicated. The further process of modification after duplication can be thought of as a divergence process of the duplicated structure. In this case, there are now multiple synchronous genes where before there was only one. We call this process a *lifting-driven duplication event*.

This mechanism provides a concise explanation for the presence of genetic fibrations, but the key point is that breaking the symmetry of duplicated circuits gives rise to a series of logic circuits. The process of duplication as lifting and eventually symmetry breaking also provides a suitable model for artificially generating an ensemble of networks with a prescribed number of fibers and broken symmetry logic circuits.

18.6 Lifting the AR base: toggle-switch or SR flip-flop circuit

By duplicating a self-repressing gene in such a way that the lifting property is respected, the duplicated circuit results in two mutually repressed genes forming a two-node negatively autoregulated (NAR) fiber (a fiber that autoregulates itself in an inhibitory manner) as shown in the first duplication event in the upper part of Fig. 18.4. This duplicated circuit is still symmetric. However, its dynamics is not the same as the negative autoregulated loop in the base, since each gene receives two inputs in the duplicated circuit, and the base receives only one. So, this duplication event does not conserve lifting, and the duplicating genes cannot be a fiber. If the genes further ‘lose’ the autoregulation, the resulting circuits is a lift of the base, the fibration is restored, and the dynamics is conserved.

When different regulators are added to each gene, the symmetry is broken, and the resulting circuit is the bistable switch or toggle-switch shown in Fig. 18.4 center. As we have seen, this circuit is analogous to a flip-flop in electronics (Tanenbaum, 2016), as shown in Fig. 18.5 with the different inputs for each gene being the *set* (*S*) and *reset* (*R*) switches.

This circuit is the bistable toggle-switch (Section 8.8.2) (Gardner et al., 2000; Leifer et al., 2020) that stores one bit of information, since it has two possible stable and reciprocal states. One of these stable states corresponds to one gene being expressed and the other inhibited; its reciprocal case is the other stable state, and the genes switch roles. Even if the symmetry is restored and both inputs return to being identical, the state of the circuit remains unchanged, storing the previous state as one bit of information. It is possible to switch between the two states by toggling the *S-R* inputs to the circuit, which is why this circuit is referred to as a *bistable* two-way switch.

The simplest form of a bistable switch corresponds to this flip-flop: the two genes inhibit each other. If other forms of self-regulation are added to this basic structure, the resulting structure still posses two different stable states, so it remains a bistable switch, albeit with different dynamics.

Such a circuit of positive feedback loops constitutes a memory storage device (Fig. 18.5). It consists of two genes mutually repressing each other (which makes it a double-negative loop). The toggle-switch exhibits two stable complementary steady states: either gene Y is expressed and represses gene Y', or vice versa. This stores one bit of binary information depending on which of the two states is present. Since these states are complementary, we are free to label one state as *Q* (say *Q* = 0) and the other as its logical opposite \bar{Q} (\bar{Q} = 1).

In the case of the flip-flop, what select the state of the circuit are the *set* (*S*) and *reset* (*R*) switches shown in Fig. 18.5. The analogous *set/reset* switches for the genetic toggle-

switch are the broken symmetry inputs: external regulators of the feedback loop that break their input symmetry isomorphisms, meaning regulators that do not regulate both A and B equally, as exemplified in Fig. 18.4. Such unequal external regulators break symmetry, causing the system to select one of the two possible states.

When the symmetry is first broken it causes the circuit to express one of the two genes. The external regulator ‘*set*’ drives the circuit to express gene Y . However, if this regulator is absent, making both S and R absent, the input tree isomorphism is restored, and the circuit remains in the same state (expressing Y). This allows the circuit to retain the memory of its previous state. Thus symmetry breaking of the circuit endows it with the ability to store memory. The state of the circuit can be toggled to the other state if instead the R external regulator is present, so that it causes the system to express Y' instead of Y . Again, if the symmetry is then restored, the circuit still preserves the memory of its previous state.

The SR flip-flop does not provide synchronized outputs. After the input signals arrive at the logic gates, each gate provides its output without waiting for the output of the other one. This results in fast oscillations which, in the application to integrated circuits in digital electronics, are undesired. In digital electronics the input $S = 0$ and $R = 0$ is therefore ‘forbidden’, since the NAND gates set both $Q = 1$ and $\bar{Q} = 1$, which violates the logical state $\bar{Q} = \text{not } Q$.

Biological realizations of the AR symmetry breaking class are shown in Fig. 18.6, both from human regulatory networks with genes *NFKB1* and *HOXA9* (upper), and the regulatory network of genes *IRF4* and *BCL6* (bottom). Gene *NFKB1* further regulates two genes, *BST1* and *HAX1* as its outputs, but this regulation does not affect the functionality of the flip-flop. The SCCs of *E. coli* are also composed of many flip-flops, more on this in next chapter.

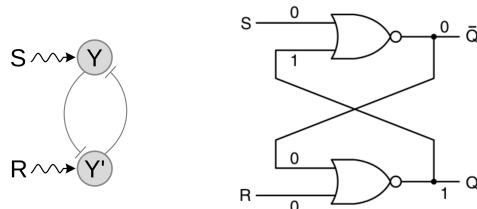


Fig. 18.5

SR flip-flop. In a genetic network, this circuit is obtained by lifting the AR base and breaking symmetry with two regulators S and R .

18.7 Lifting the FFF base: clocked SR flip-flop circuit

The process just described corresponds to breaking symmetry in an $|n = 1, l = 0\rangle$ circuit with inhibition loops. An analogous duplicating-by-lifting symmetry breaking process for the circuit generated from an FFF building block base, shown at the bottom of Fig. 18.4, gives a circuit resembling a clocked SR flip-flop (Leifer et al., 2020; Horowitz and Hill, 2015;

Tanenbaum, 2016), which basically acts as a more complex flip-flop or toggle-switch. This process can be repeated with more complex fiber building blocks, following the hierarchy of complexity depicted in Fig. 18.6.

As shown in the middle column of Fig. 18.6, the X and Y genes from the FFF base are lifted to create a circuit isomorphic to the Clocked SR flip-flop. The base contains one external regulator X that is also lifted to X and X' so that the S-R inputs are received by these additional regulators along with the signal from an extra ‘clock’ gene. The symmetry is broken by regulators or inducers of genes X and X' acting as S (set) and R (reset) of memory, and it is restored when S = R, leaving the system ‘magnetized’. This new circuit is the clocked SR flip-flop in which the effect of the S-R inputs is now conditional on the presence or not of the *clock*.

Lifting the FFF adds a second level of NAND logic gates to the SR flip-flop via gene X. In order to have consistent logic operations, an input clock gene CLK is added which symbolizes the activation of gene X, since gene X needs to receive input for its activation.

The second level flip-flop inverts the outputs of the previous SR flip-flop logic circuit, meaning that when $S = 1$ and $R = 0$ ($S = 0$ and $R = 1$), the circuit outputs $Q = 1$ and $\bar{Q} = 0$ ($Q = 0$ and $\bar{Q} = 1$). The input $S = 0$ and $R = 0$ results in an unchanged state. When the input of gene CLK is low, $CLK = 0$, the output of the second level of both NAND gates outputs high signals, independently of the values of S and R, assuring that the outputs Q and \bar{Q} remain unchanged. However, when the clock input is $CLK = 1$, it allows the first level of NAND gates to change the outputs for $S \neq R$. The clocked SR flip-flop also has a forbidden state when $S = 1$ and $R = 1$ in digital electronics.

In Fig. 18.6, we show two biological realizations of the Clocked SR flip-flop: the set of genes {CLOCK, NR0B2, NR3C1, E2F1 and TP53} and the set {CEBPB, DDIT3, PRDM1, CEBPA and MYC}. Interestingly, there is a gene called CLOCK exactly at the clock position in this flip-flop. This flip-flop turns on and off the activity of TP53, a major tumor protein that is mutated in most of human cancers. Both examples are from human genetic regulatory networks. The outputs of the flip-flops, like E2F1 and TP53, further regulate a set of genes each as indicated by the red genes in the figure. These circuits are also abundant in *E. coli*, see next chapter.

18.8 Lifting the Fibonacci base: JK flip-flop circuit

Adding now an extra edge feeding information back into the regulator of the FFF gives a *Fibonacci* building block, described in detail in Section 15.5, an additional step in the complexity hierarchy of broken symmetry circuits. As for the FFF broken symmetry circuit, the lifting of the Fibonacci base results in a circuit analog to the *JK flip-flop*, the most widely used of all flip-flop designs (Horowitz and Hill, 2015). It is a universal flip-flop since it can be configured to perform as any other flip-flop. The JK flip-flop is basically an SR flip-flop with a longer feedback loop. In its symmetric state it acts as a toggle command that changes the output to a logical current value.

In its symmetric state, the JK flip-flop is isomorphic to the symmetric Fibonacci base.

In the symmetry broken phase, it acts as a memory device which presents two possibilities (Fig. 18.6 third column). A ‘chiral’ symmetry (where Y feeds X and Y’ feeds X’) and a ‘parity’ symmetry (left-right reflection, where Y feeds X’ and Y’ feeds X). This last one is the one realized in biological circuits, see Fig. 18.6, third column. Examples of JK flip-flops are abundant in gene regulatory networks in humans and mice.

These more complex circuits were not observed in the bacterial TRN studied in this work but in more complex species like yeast and human. Two examples of the Fibonacci broken symmetry circuits are shown in Fig. 18.6 for the sets of genes {*PITX1, JUN, NKX3-1, TP53* and *ESR1*} and {*FLII, HDAC1, EPS300, AR* and *RELA*}, both from human genetic networks. We also show the set of genes regulated by these flip-flops.

The external regulator genes, J and K, provide inputs that are logically processed by the circuit according to the type of interaction links between the genes: activators (black arrows) or repressors (red flat links). The outputs of the circuits (green genes) regulate the expression levels of other genes (in red) without affecting the circuit functionality. Gray arrows correspond to interactions with unknown functionality.

These are wonderful examples of the TRN computational core. Crucial to the dynamics of these circuits are feedback loops between different genes. Their presence implies that the circuits are always embedded into the SCCs of the network. It means that we can interpret the SCCs of the network as the modules where the logical computations are performed. This process can be extended to higher complexity fiber building blocks to obtain ever more complex broken symmetry circuits.

18.9 The TRN as a computer

This notion of genetic circuits performing basic logic computations becomes important when trying to understand the TRN in bacteria, discussed at greater detail in Chapter 19. The TRN can be understood as a computational machine in charge of computing the decision-making of the bacterium. Kondev (2014) has put this idea very clearly: for bacteria (and us), it is all about ‘To eat or not to eat sugar’. This binary decision is made by an internal digital computer embedded in the TRN.

The core driver of the TRN is an ensemble of logic circuits containing both memory storage devices resembling toggle-switches, and timekeeping devices in the form of genetic oscillators. This is of relevance, given that from the perspective of modern computational devices any computer needs at the most basic level these two modules: a memory device component and a timekeeping component. This motivates the question of how we can understand the bacterium’s decision-making process as a logical computational process, and to understand a bacterium as a computational device. In the particular case of *E. coli*, we find negative feedback loops as oscillators, and both toggle-switches and FFF broken symmetry circuits. However, we have not found Fibonacci broken symmetry circuits which appear in more complex species like humans. More in-depth discussion is given in Chapter 19.

In order to fully understand the dynamics of this circuit it is crucial to understand their

external inputs and how are they embedded in or connected to the rest of the TRN. This complements what has been done in their implementation and design by the synthetic biology community. On the one hand, external inputs produce the symmetry breaking that allows some of the circuits to store memory; on the other hand, understanding how these circuits are embedded in the full network, and which functional modules of the TRN communicate with each other, and how, are central to understanding bacterial processes as logical computational ones.

18.10 Algorithm to find fibration symmetry breaking circuits

To find the circuits mentioned above, we run the algorithm developed by Leifer et al. (2020) looking for the induced subgraphs of the network whose connectivity is identical to the logic circuits we are looking for. The algorithm is explained in more detail in the Supplementary Information of (Leifer et al., 2020). Corresponding codes can be found at <https://github.com/makselab/CircuitFinder>.

Briefly, the algorithm consists of two steps: (1) Find the subgraphs of the minimal network isomorphic to a broken symmetry building block, and (2) Keep only those subgraphs that are induced. Some of the observed circuits, however, have a slightly different topology due to self-regulation of some genes in these circuits, so their dynamical behavior might be different.

18.10.1 Software to find broken symmetry circuits

Leifer et al. (2020), supplementary section ‘Algorithm to find broken symmetry circuits’, explains the algorithm to find broken symmetry circuits. The internal dynamics of each node in the network is defined by its inputs, therefore the internal dynamics of the circuit is defined by its topology. We believe that external inputs to all the nodes of the circuit can switch the state of the circuit, but cannot change the function that it performs. Therefore, the circuit we are looking for is a circuit that has a given structure, with the possibility of external inputs. Hence, the required algorithm is one that can find all the circuits with given topology ‘embedded’ in a network. Mathematically, this is expressed as the search for an *induced subgraph* of a graph, see Definition 3.6.

To count the number of occurrences of broken symmetry circuits in a given graph we count the number of appearances of induced subgraphs with the given topology. The problem of finding broken symmetry circuits consists of three steps:

- Create a duplicate by lifting the symmetry circuit.
- Find subgraphs isomorphic to the replica symmetry circuit.
- Remove subgraphs that are not induced.

The first step is described in detail in Section 18.3. The second step is to find subgraphs isomorphic to the lifted symmetry circuit. The general idea is to choose a subgraph and

check whether it is isomorphic to the circuit concerned, and to continue doing this for all possible subgraphs of the network. However, this straightforward approach is computationally expensive. For example, two of the broken symmetry circuits in Fig. 18.7 consist of 5 nodes. The computational time required to check this circuit is N^5 , where N is the total number of nodes in the graph, which means that for big enough graphs the problem becomes computationally infeasible. The same problem occurs in the search for motifs.

Different approaches to this problem and applications to graph matching and pattern recognition have been widely studied and are reviewed in (Conte et al., 2004). Time costs can be cut if unprofitable paths are identified and skipped in the search space. One of the recent works in the field is the VF2 algorithm developed by Cordella et al. (2004). It is designed to deal with large graphs and uses state-of-the-art techniques in order to reduce computational time. In our code, we use the algorithm implemented in a popular R package igraph (Csárdi and Nepusz, 2006) as a function *subgraph_isomorphisms(...)*.

The third step is to remove all subgraphs that are not induced or, simply speaking, have extra links between the genes in the broken symmetry circuit. Our procedure is: take a node set identified above, find the induced subgraph of the complete graph with this node set, and compare the adjacency matrix of the induced subgraph with the adjacency matrix of the circuit (modulo permutation). If the matrices are different, then the circuit is removed. All remaining circuits are the broken symmetry circuits. By agreement, multi-links and self-loops are removed from the network prior to consideration.

By applying the steps described above, we get the full list of induced subgraphs that are isomorphic to the given circuit, which we take as broken symmetry circuits. The implementation of this code can be found at <https://github.com/makselab/CircuitFinder>.

A few biological realizations of this symmetry-breaking process are shown in Fig. 18.6, last row. Leifer et al. (2020) perform a systematic search for these circuits using the algorithm above. The full list of symmetry broken circuits in gene regulatory networks across species appears in the Supplementary Materials of (Leifer et al., 2020). Table 18.1 shows the Z-scores of these circuits indicating their high significance.

Species	Database	Nodes	Edges	N_{real}	SR flip-flop $N_{\text{rand}} \pm SD$	Z-score
Arabidopsis thaliana	ATRM	790	1431	47	1.6 ± 1.2	36.40
Micobacterium tuberculosis	Research article	1624	3212	6	1.7 ± 1.4	3.20
Bacillus subtilis	SubtiWiki	1717	2609	3	2.1 ± 1.4	0.6
Escherichia coli	RegulonDB	879	1835	14	2.1 ± 1.4	8.40
Salmonella SL1344	SalmoNet	1622	2852	6	1.4 ± 1.2	3.80
Yeast					27	
	YTRP_regulatory	3192	10947	9	5 ± 2.5	1.60
	YTRP_binding	5123	38085	31	21.6 ± 5.8	1.60
Mouse	TRRUST	2456	7057	82	4 ± 2.1	37.70
Human					192	
	TRRUST	2718	8215	89	4.3 ± 2.1	40.50
	TRRUST_2	2862	9396	103	5 ± 2.3	43
Species		Clocked SR flip-flop		N_{real}	JK flip-flop $N_{\text{rand}} \pm SD$	Z-score
		$N_{\text{rand}} \pm SD$	Z-score		$N_{\text{rand}} \pm SD$	
Arabidopsis thaliana		3	0.2 ± 0.5	5.80	2	0 ± 0
Micobacterium tuberculosis		0	N/A	N/A	0	N/A
Bacillus subtilis		0	N/A	N/A	0	N/A
Escherichia coli		3	0.3 ± 0.8	3.30	0	N/A
Salmonella SL1344		0	N/A	N/A	0	N/A
Yeast					1	
YTRP_regulatory		3	3 ± 3.6	0	0	N/A
YTRP_binding		192	103.3 ± 45.6	1.90	2	6.8 ± 6.1
Mouse		216	1.9 ± 2.7	79.50	25	0.004 ± 0.06
Human					90	
TRRUST		247	3.5 ± 4.8	50.60	45	0.02 ± 0.2
TRRUST_2		319	5.9 ± 7.2	43.20	45	0.02 ± 0.3

Table 18.1: **Lifting-induced duplication circuits by fibration symmetry breaking are statistically significant over many networks and species.** We report the corresponding Z-score statistics as computed in Table 17.2 for the symmetry circuits.

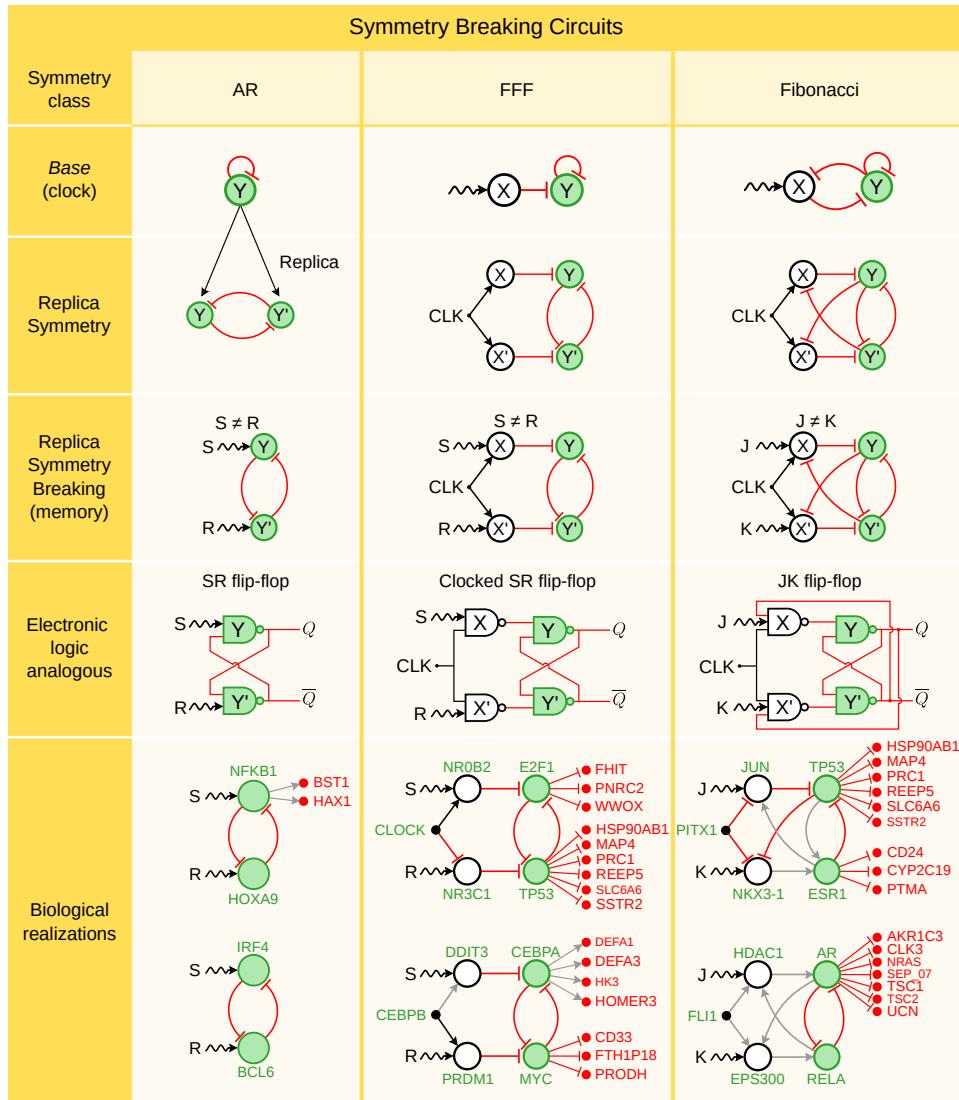


Fig. 18.6

Broken symmetry hierarchy. First column: **AR**. The lifting-driven duplication of the AR circuit results in a circuit analogous to the SR flip-flop. The symmetry between Y and Y' is broken by the inputs S and R , such that $S \neq R$, resulting in a pair of logical outputs Q and $\bar{Q} = \text{not}(Q)$. Second column: **FFF**. Following the same strategy, we lift the FFF base through duplication. This operation adds a second level of logic gates to the SR flip-flop. Third column: **Fibonacci**. The lifting-duplication of the Fibonacci base results in a logic circuit analogous to the JK flip-flop. Figure reproduced from (Leifer et al., 2020). Copyright © 2020, Leifer et al.

a Clocked SR flip-flop b JK flip-flop

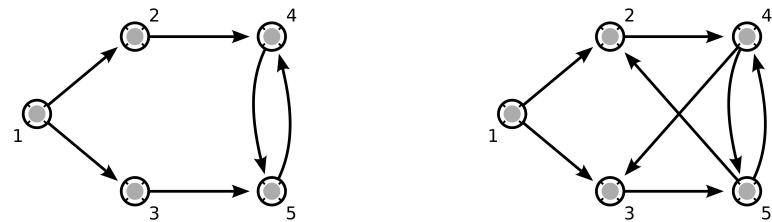


Fig. 18.7

Examples of the broken symmetry circuits. (a) Clocked SR flip-flop. (b) JK flip-flop.

Taming Biological Complexity with Symmetries: the Minimal Genome

We show that the TRN can be reduced to its minimal components, or ‘minimal genome’—an effective minimal base network obtained by the application of fibrations. The central maps used for dimensional reduction of the TRN are two fibrations: the minimal surjective graph fibration and the inverse injective fibration. The entire biological network is distilled into a machine that performs computations over time. This computation is executed by a set of flip-flops, which are symmetry-breaking circuits discussed in Chapter 18, along with clocks and synchronized fibers, which are symmetric circuits explored in Chapters 14 and 16. In this system, part of the network functions as a memory device, while another part operates as synchronized clocks. Consequently, each gene is assigned a specific function within the regulatory machine, allowing the network to be reduced to its minimal functional form.

19.1 Fibration complexity reduction

The sheer complexity of biological networks can often make it challenging to decipher the underlying mechanisms that govern their behavior. Traditional methods for analyzing biological networks, such as graph theory and differential equations, often result in large, unwieldy models that are difficult to interpret and analyze. These models can also suffer from issues such as over-fitting, where the model is too complex and thus unable to generalize to new data, or under-fitting, where the model is too simple and thus unable to capture the nuances of the underlying biology.

To overcome these challenges, researchers have developed a variety of methods for complexity reduction in biological networks. Such techniques aim to simplify the network while retaining its essential features, making it easier to analyze and understand. These methods range from simple heuristics such as filtering out low-confidence interactions, to more sophisticated techniques such as clustering and dimensional reduction through machine learning techniques.

One of the most interesting applications of our new notion of symmetry is to the simplification of biological systems, particularly networked biological systems, by reducing the complexity of the system while still preserving its key dynamical features. For instance, describing the set of ODEs underlying gene expression dynamics in a TRN requires knowing a large number of microscopic parameters governing gene input functions. These parameters capture the entire process from transcription, translation, protein folding to mRNA and protein degradation, including details such as binding and unbinding to DNA, ribosome

and polymerase binding, mRNA and protein lifetimes, etc. (Klipp et al., 2016). This complexity inherent to the large dimensionality of the multiparameter space is common to all processes in biology. Finding low-dimensional effective models to describe the dynamics is crucial for understanding how function and behavior in biological systems emerge from their complex underlying microscopic dynamics.

In physical systems, very often we are also confronted with high-dimensional models in order to fit experimental data. However, in physics, well defined theoretical models have often been developed, which allow the definition of theoretical descriptions of complex phenomena with few or no free parameters to tune, using the underlying symmetries of the system. For example, the Standard Model of particle physics, with only (!) 18 parameters (Cahn, 1996) can describe all fundamental particle interactions (with the exception of gravity) with extremely high precision. Renormalization group techniques are used to ‘smooth out’ the details of the system, focusing on the effective ‘larger picture’ models that arise at critical points (liquid-gases transition, superconductivity, superfluidity, phases, etc.). These results led to the idea of universality in critical phenomena where simple macroscopic properties of the system emerge from their microscopic parameters while being independent of them. With this, the dimensionality of complex systems is effectively reduced by the underlying symmetries of the system, while still preserving the fundamental properties. This makes symmetry principles the fundamental cornerstone of most physical laws.

In the biological sciences, however, a *systematic* way to perform dimensional reduction is still lacking, despite a large amount of work (Stephens et al., 2011). Álvarez-García et al. (2025a) address the issue of complexity reduction in biology for the TRN of *E. coli* by reducing the network to its minimal computational apparatus. This dimensional reduction is derived in two main steps: first collapse genes with isomorphic input trees using surjective fibrations, which we detail below; then identify the core driver subnetwork which drives the dynamics and logic computations of the entire system with injective fibrations and k -core decomposition.

19.2 CoReSym: reducing the TRN to its minimal computational core

Álvarez-García et al. (2025a) introduce a scheme, called Complexity Reduction by Symmetry or *CoReSym*, to reduce a gene regulatory network to its *minimal genome* in a way that preserves its dynamics and uncovers its computational capabilities. Figure 19.1 shows the reduction procedure for the case of *E. coli*: the TRN on the top is reduced to the simpler, more comprehensible structure on the bottom, which elucidates how the network works.

Below we elaborate on how to use the fibration machinery to obtain such a minimal genome in bacteria. The dimension reduction process consists of five steps:

- Step 1. Collapsing: graph reduction via symmetry fibrations, by collapsing all the redundant edges.

dant symmetric pathways. This represents a reduction of 70% of the original *E. coli* TRN.

- Step 2. Pruning: reduction via k -core decomposition, analogous to an inverse injective fibration. This step removes nodes like enzymes that do not regulate other genes directly. This represents a further reduction to only 2% of the original *E. coli* TRN.
- Step 3. Identification of the Minimal TRN: breaking down the core minimal network into strongly connected components (SCC).
- Step 4. Computational core: Identification of symmetry breaking building blocks in the Minimal TRN.
- Step 5. Simple cycles: Identification of simple cycles in the Minimal TRN's SCCs.

Computationally, the steps in CoReSym consist in searching for fibration building blocks (Morone et al., 2020), symmetry breaking building blocks (Leifer et al., 2020) and cycles in the core network (Purcell et al., 2010).

Steps 1 and 2 are concerned with the reduction of the network, removing all elements in it that do not contribute to its computational capabilities (Fig. 19.2). This reduces the TRN to the core network at the heart of the decision-making processes: the minimal TRN. Step 1 eliminates all information pathway redundancies. Step 2 removes the nodes that only receive signals or pass through it without contributing to the decision-making process; these nodes are responsible for communicating the output from the minimal TRN to the periphery. Steps (3)-(5) analyze this minimal TRN. In step (3) we focus on the large-scale structure of the minimal TRN: how the components of this core network are connected to each other. The last two steps ‘zoom in’ to look at the small-scale, or local, structures within the minimal TRN’s components, by looking at the logical circuits (step 4) and how these are connected with each other, as well as providing information on the connectivity structure within the different components (step 5).

Step 1 and 4 decompose the genetic network into its building blocks via fibration symmetries and broken symmetries, respectively. Together, these steps lead to the identification of the function for every single gene in the TRN as belonging to three general classes of genes:

- (1) A set of synchronized symmetric fibers following the hierarchies of Chapters 14 and 16
- (2) Regulators of the SCCs
- (3) Broken fibration symmetry circuits within the SCCs following the hierarchy of Chapter 18. These can be further classified into:
 - (3.1) memory devices (flip-flops or toggle-switches): broken symmetries of the AR, FFF, Fibonacci and $n = 2$ fibers.
 - (3.2) oscillators.

The synchronized nodes obtained by symmetry fibration analysis are characterized by five basic type of fibers, Fig. 18.3:

- (1) Trivial fibers made of regulons, operons, etc., with no loops within the fibers; they are $n = 0$ fully feed-forward. They make up 67% of the building blocks in *E. coli*.

- (2) Feed-forward fibers making up 26.4% of *E. coli*'s building blocks. They are subdivided in sub-classes according to the number of external regulators:
 - (2.1) 0-FFF, 14%,
 - (2.2) 1-FFF, 9%,
 - (2.3) 2-FFF, 3%.
- (3) Fibonacci fibers φ -FF with fractal dimensions $\varphi = 1.61, 1.31$, and 1.15 , representing 3.3% of the building blocks.
- (4) $n = 2$ fiber (2.2%).
- (5) Composite multilayer fibers: a composition made of the previous ones (1.1%).

These building blocks are arranged in a large-scale structure made of SCCs with a rich cycle structure, assembled into an effective feed-forward structure that we call the *minimal TRN*. This structure contrasts with previous representations of the bacterial TRN as a purely feed-forward network (Ma et al., 2004; Martínez-Antonio et al., 2008; Dobrin et al., 2004; Shen-Orr et al., 2002).

Name	Description
Full Network	879
In Fibers	416
Base	555
Minimal (1-(out)core)	42
In SCCs	24
1st type regulators	11
2nd type regulators	4
3rd type regulators	2

Table 19.1: **Count of genes in the dimensional reduction of *E. coli* TRN.** Table reproduced from (Álvarez-García et al., 2025a).

We use the TRN of *E. coli* from RegulonDB (Gama-Castro et al., 2016) and we treat it under the idealized assumptions stated before in Sections 5.3. Most important for this purpose is the ‘uniform broadcast’ idealization explained in detail in Section 21.5: for genes within a fiber, a) their input functions depend only on the transcription factor and not on the target gene’s binding site, which is mostly satisfied, and b) that their constants for degradation and maximal synthesis rates are equal across the fiber’s genes. As explained in Section 15.8, we take an operon to be a single node in the TRN, which removes this set of trivial fibers from the analysis. This means that our initial global TRN is made up of 879 nodes, see Table 19.1.

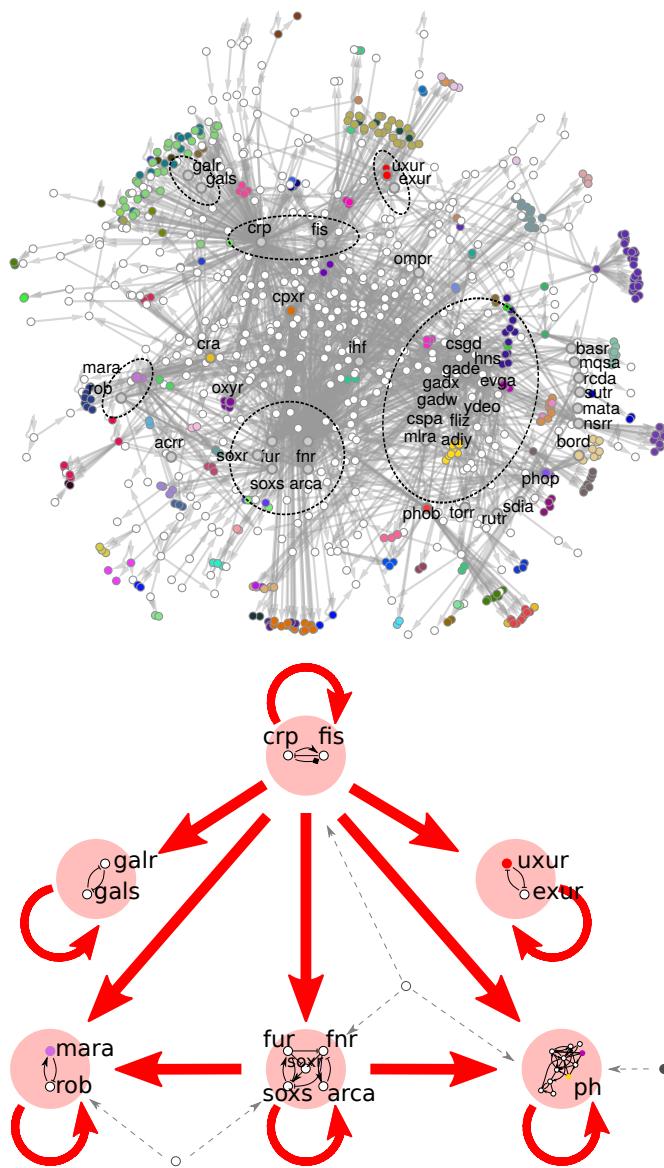
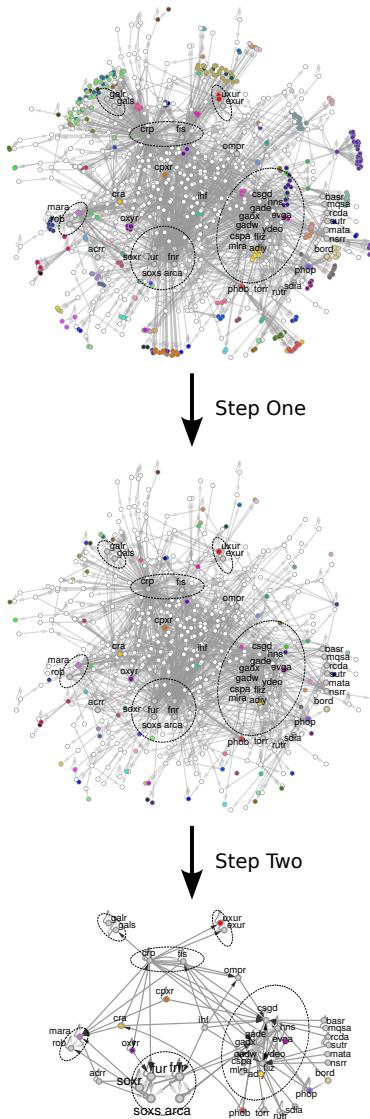


Fig. 19.1

The effective minimal TRN obtained through CoReSym. Top: The weakly connected component of the 879-node operon-TRN for *E. coli*: sizes of nodes and font size are proportional to the out-degree of the node. Bottom: The effective minimal TRN obtained after the application of CoReSym, illustrating the signal flow between its different components (bigger red nodes), which are the strongly connected components (SCCs) of the network. Figure reproduced from (Álvarez-García et al., 2025a).

**Fig. 19.2**

Steps of CoReSym. We start with the full network. SCCs are marked by ellipses; names shown are only for the genes belonging to the core of the network, obtained after reduction. Various nodes with notably high out-degree do not belong to the core network. Step 1 of our procedure collapses all fibers into one representative node, resulting in the base network of the symmetry fibration. The remaining nodes maintain their positions. Step 2 removes all null-paths ending at nodes with no output via the k -core decomposition, resulting in the core of the network with only 42 nodes. Figure reproduced from (Álvarez-García et al., 2025a).

19.3 Symmetry fibrations reduce the network yet preserve information flow

By considering ‘message passing’ between genes by means of transcription factors in the TRN, we understand this network as an information-processing network. We simplify this network using fibration symmetries. In terms of the flow of information, fibers define redundant pathways of information, since all genes in a fiber receive exactly the same information. Collapsing the fibers then reduces the network considerably, while still preserving the dynamics in the TRN, given that we are eliminating only redundancies in the flow of information without eliminating any information pathway. The information flow is the feature we wish to preserve, in order to understand the decision-making processes of bacteria. The maximum reduction corresponds to the symmetry fibration that reduces the network to its minimal base, a part of the minimal genome. This step accounts for a great reduction in the number of genes present, given that out of the 879 genes, 416 of them belong to 92 fibers, see Table 19.1.

19.4 Finding the driving core

The second step in the reduction finds the core of the network that is responsible for driving the dynamics of the global TRN. This corresponds to performing an inverse injective fibration.

19.4.1 Injective fibrations

An injective fibration formalizes the intuitive notion that under certain conditions the dynamics of an entire system may be driven by only a subset of its constitutive elements. This is formalized by (DeVille and Lerman, 2015b, Lemma 5.2.1), where it is shown that for an injective (one-to-one) fibration $\varphi : G \rightarrow G_2$ the dynamics of the bigger graph G_2 is driven by the dynamics of the smaller graph G . In fact G is (isomorphic to) a subgraph of G_2 . This can be seen in Fig. 19.3 for the injective fibration from G to G_2 .

The emphasis here is on the injective nature of the map φ . For a mapping to a bigger graph to be a fibration, like the one shown in Fig. 19.3, it must satisfy the lifting property. This requires that each edge in G_2 whose target node is an image from a node in G (i.e. 1', 2', 3', 4'), can be lifted to a unique edge in G . This implies that no new edges are allowed to target any of the nodes of the original graph (i.e. 1', 2', 3', 4'), satisfying the lifting property.

As a consequence, all the added nodes in G_2 (i.e. 5', 6', 7') must strictly be targets only of the image nodes. Hence signals flow only outwards from G , and so the dynamical state of the outer nodes is driven by the dynamics of the original smaller graph G . In other words,

the subgraph G of G_2 drives or controls the entire dynamics of G_2 . This, in turn, guarantees that the dynamics of the original graph is preserved in the image graph G_2 .

DeVille and Lerman (2015b) show that when a fibration $\varphi : G \rightarrow B$ is injective, the domain G corresponds to the subnetwork of the system driving the dynamics of the entire system B , the codomain. Since φ is a fibration, all nodes i' in B that are not image nodes must be targets of image nodes and must not send information back to G ; so the information flows only outwards of G . In our case, the problem we face is going in the opposite direction to the map: from the entire system we want to find the subsystem responsible for its dynamics, meaning we want to construct an inverse of the injective fibration.

19.4.2 k -core decomposition

Therefore we need to determine the nodes that only receive information, or that only pass information along, since these are clearly not responsible for driving the dynamics. This is done by the *k -core decomposition* of the network (Kitsak et al., 2010). This decomposition ‘peels off’ layers, formally called *k -shells*, from the network, Fig. 19.3c, d. This is achieved by assigning a *coreness index* k_s to each node, corresponding to the respective shell each node belongs to. The k -shell corresponds to the set of nodes with coreness k_s .

The smaller k_s is, the more peripheral the node is, as can be seen in Fig. 19.3c. The coreness of a node captures the degree of the nodes it is connected to. For example, node b in Fig. 19.3c has a smaller coreness ($k_s = 1$) than node a ($k_s = 2$), even though b has a higher degree of 5 compared to a ’s degree of 3. This is because b is connected mostly to nodes with degree 1, in contrast to a , which connects to nodes with higher degree, making it more *influential* (Kitsak et al., 2010) or central.

Definition 19.1 k -core and k -shell of the network. The *k -core* of a network is the maximal induced subgraph, subject to the condition that the number of edges of every node within the k -core is at least k . The coreness of each node is then the k -shell number k_s for which the node belongs to the k_s -core but not to the $(k_s + 1)$ -core (Fig. 19.3c, d).

Since the TRN corresponds to a directed network, every node has an in-degree and an out-degree. However, the out-degree is not relevant here, since the network formalism we are using represents all variables involved in the dynamics of a node by the source nodes of its input set. Thus we iteratively eliminate the nodes with no out-degree, which corresponds to the $k_{\text{out}} = 0$ shell. This means that all paths that end at a node with no output are removed, so what remains are the paths that loop back to some previous node in the path. Because of this, all the strongly connected components of the network are still present. The result is the $k_{\text{out}} = 1$ core of the network.

The order in which these two steps are applied matters. If we find the $k_{\text{out}} = 1$ core before eliminating redundancies in information pathways, the resulting network might still have redundant pathways that are not actually computing anything relevant, only transmitting information outward. These pathways should not be part of the core driver of the network. In particular, given a 2-node SCC whose nodes are also symmetric (belong to the same fiber), if both nodes have out-degree at least 1 they still belong to the $k_{\text{out}} = 1$ core. However,

if we apply the symmetry fibration first, these nodes collapse to a single node with only a self-loop, meaning that they may no longer be part of the $k_{\text{out}} = 1$ core. By definition the 1-shell of the TRN is made of enzymes and other proteins except for TFs. A TF cannot be part of the out-going 1-shell, since a TF always regulates at least one other gene, so it has out-degree 1 or more.

Most of the 1-shell then comprises enzymes that catalyze metabolic reactions. When we consider a TRN, these enzymes send information to the metabolic network. This information can then return to the TRN since metabolites and small molecules can bind to the TF to activate or inhibit their activity. We do not consider this important type of regulation here, so the outgoing 1-shell can be removed from the network for now. An integrated biological network should include this loop of information by considering couplings between the TRN and the metabolic networks, as well as the protein-protein interaction network.

After this decomposition, the network is reduced from 555 nodes to only 42 (see table 19.1), which correspond to the driver core of the TRN. The dynamics of the global TRN then propagates outward from this minimal TRN. Table 19.2 shows how the number of nodes decreases at each step of CoReSym. Table 19.3 shows the coverage.

19.5 Structure and composition of the minimal TRN: a simple computer

The next step in the reduction of the TRN is to calculate the strongly connected components.

The overall structure of the reduced TRN, the computational minimal core of *E. coli* TRN corresponds to a feed-forward structure between the SCCs (with autoregulation loops

Reduction step	Gene count	%
Step 0.0: Full Genome RegulonDB	4,690	-
Step 0.1: TRN (non isolated TFs)	1,843	100%
Step 0.2: operon-collapsed TRN	879	48%
Step 1: Base-TRN (collapsing fibers)	555	30%
Step 2: Minimal TRN (after trivial pruning)	42	2%

Table 19.2: **Reduction count and fiber coverage.** The full *E. coli* genome contains 4,690 genes, of which 1,843 genes express TFs that are not isolated, i.e. they are TFs with at least one regulation to or from another TF. The first reduction of operons is performed by a trivial fibration that collapses the operons that are trivial fibers. This is in reality just a part of Step 1 and reduces the network to the nontrivial TRN with 879 TFs, which is then used as the initial step of the nontrivial reduction process. The fibration reduces the network to 30%, and after removing the trivial dangling ends the final base is reduced to 2% of the original size, constituting the minimal TRN. Table reproduced from (Álvarez-García et al., 2025a).

within them), with the information flowing outward from the *crp-fis* SCC, the most central master regulators genes in *E. coli*, illustrated in Fig. 19.1 bottom. In addition to this feed-forward structure, the reduced TRN is completed by regulator nodes that connect the SCCs to each other, and also feed information directly into them. We obtain 6 SCCs (Fig. 19.4, three of them already discussed in Section 14.3); one with 11 nodes which is in charge mostly of the pH and stress response of the cell (pH SCC); one with 5 nodes (the *soxS* SCC); and the rest with only two nodes (*crp-fis*, *marA-rob*, *galR-galS* and *uxuR-exuR*). The *crp-fis* SCC works as a type of master regulator, controlling the rest of the SCCs and being regulated by only two external outputs: the *cra-fiber* and the *ihfAB* TF. The reduced TRN can also be understood as an ensemble of simple genetic logic circuits that perform basic logical computations, like those explained in the previous section.

19.5.1 The minimal TRN

The minimal TRN network is defined as follows:

Definition 19.2 Minimal TRN network (Álvarez-García et al., 2025a). The *minimal TRN* is an effective network in which each SCC is a super-node; that is, a subnetwork that can be considered to be a node in some related network. A directed edge between two supernodes means that there is at least one directed edge between a node of the corresponding SCC and a node in the SCC at the target of the edge. (This is a standard construction in graph theory, called the *condensation* of the graph.)

Figure 19.4 shows the application of the fibration to each of the SCCs (see also Figs. 14.3, 14.4 and 14.5) and the minimal TRN of *E. coli* is shown in Fig. 19.1 (bottom).

The minimal TRN is a feed-forward structure without cycles, only loops at the supernodes. At the root is the carbon SCC, which works as a type of master regulator, controlling the rest of the SCCs. All these SCCs are regulated by different genes. For instance the *cra-fiber* and *ihfAB* regulates the carbon SCC.

The importance of the carbon *crp-fis* SCC is exemplified by its central position in the minimal network. This two-node SCC also regulates the biggest fibers, since the main

operon-TRN breakdown	Gene count	%
operon-TRN	879	100%
Nodes in fibers	416	47.3%
k_{out} shell (not in fibers)	431	49%
Nodes in SCCs (not in fibers)	20	2.3%
Connectors (not in fibers)	12	1.4%

Table 19.3: **Breakdown of the operon-TRN network.** Step 1 collapses the 416 nodes in fibers into just 92 fibers, to give the Collapsed-fibers 555-node network. Step 2, removes all the Shell nodes and 82 of the fiber-collapsed nodes, leaving only the 42 nodes in the Minimal TRN of *E. coli*. Table reproduced from (Álvarez-García et al., 2025a).

function of the bacterial cell is to process sugars. Also of interest is that there are some fibers that are jointly regulated by two SCCs, and that the *uxuR-exuR* and *galR-galS* do not regulate any fibers of their own, although they do regulate other genes and operons.

The *galR-galS* SCC and the *uxuR-exuR* SCC receive signals from *crp-fis*, but do not receive or send signals to the rest of the SCCs. They compute their state solely based on *crp-fis* SCC's input and then send their corresponding outputs to the genes that they regulate. The *crp-fis* pair acts as a toggle-switch that turns on and off, depending on environmental clues.

The other SCCs are arranged in the shape of a feed-forward motif: *crp-fis* SCC feeding the *soxS* SCC and the *pH* SCC, with the *pH* SCC also receiving from the *soxS* SCC and another similar structure with the *marA-rob* SCC in the same role as the *pH* SCC: receiving from both *crp-fis* and *soxS*. The overall structure of the computational minimal core of the *E. coli* TRN corresponds to two feed-forward structures between the SCCs.

Inside each SCC there is a bunch of toggle-switch circuits which can be seen as broken symmetry circuits from fibration symmetries that provide the network with its computational capacity. Most of the oscillators and synchronized fibers, which are purely symmetric circuits, are controlled by these switches.

Figure 19.5 shows all the different possible configurations of building blocks observed in both *E. coli* and *B. subtilis* by Álvarez-García et al. (2025a). A computational TRN can be now assembled from such a circuits.

19.5.2 Toggle-switches (flip-flops) in the TRN

Inside the SCCs of the TRN we found a variety of genetic circuits with broken symmetries. In total 12 different pairs of genes were found to be involved in a number of logic circuits; see Fig. 19.6 for a depiction of the full symmetry breaking for all electronic circuits in *E. coli*.

The *galR-galS* and the *uxuR-exuR* SCCs correspond to circuits almost identical to toggle-switches except for their self-inhibitory loops which compute their state based on the *crp-fis* SCC's state, and then send this information outward to the fibers that they regulate. The *uxuR-exuR* SCC forms a nicely regulated 'almost toggle-switch': when *crp* is active it seems to induce the SCC to the state of *uxuR* being expressed, and a very similar mechanism applies to the *galR-galS* SCC, with *galR* being active. Both of these circuits present additional negative autoregulations in each gene (see Fig. 19.6), so their dynamics varies compared to the classic toggle-switch design, but their switching function remains the same.

The pair *uxuR-exuR* cycle with the regulations from *crp* is the *Mutual Repression Network with Negative Autoregulation* studied by Hasan et al. (2019). This circuit can show two distinct stable states, and therefore serves as a memory; when *crp* is active, it can induce a state in which *uxuR* is expressed while *exuR* is repressed.

The third toggle-switch-like circuit is between *csgD-fliz* in the *pH* SCC, with numerous possible ways for its symmetry breaking to occur. This circuit contains a positive autoregulation. As shown in (Leon et al., 2016), this allows for two stable states, making it possible for it to function as a memory device.

The remaining 3 SCCs are arranged in the effective minimal TRN (Fig. 19.1) as an FFF-like effective building block as an overarching structure: *crp-fis* SCC feeding the *soxS*

SCC and the *pH* SCC, with the *pH* SCC also receiving from the *soxS* SCC and another similar structure with the *marA-rob* SCC in the same role as the *pH* SCC, receiving from both the *crp-fis* and the *soxS* SCC. If we add an effective AR loop to each SCC, symbolizing that the information also travels inside the SCC, the result somewhat resembles an FFF at the global level of the whole network.

Overall, we observe six circuits almost identical to toggle-switches, three being an SCC (*galR-galS*, *uxuR-exuR* and *crp-fis*), while the other three form part of the *pH* SCC (*fliz-csgD*, *fliz-gadW* and *gadW-gadX*). Based on their inputs, *galR-galS* and *uxuR-exuR* may possibly develop only one state: *galS* and *uxuR* active, respectively. The remaining ones seem capable of both stable states, given the numerous possible ways for their symmetry to break. Most of these states would be only transient, because of self-inhibitory loops for most genes, with the exception of the states where *crp*, *csgD* or *gadX* are active in their respective circuits, given their self-activation.

19.5.3 Oscillators (clocks) in the TRN

For oscillator-type circuits we observe eight circuits: *crp*↔*fis*, *rob*↔*marA*, *soxS*↔*fur*, *fnr*↔*arcA*, *gadX*↔*hns*, *cspA*↔*hns*, *gadW*↔*gadX*, *gadX*↔*gadW*. Most of the observed oscillators are known to have damped oscillations; however, *gadX*↔*gadW*, *csgD*↔*gadW* and *crp*↔*fis* may have the same topology as the more robust Smolen oscillator.

Four of them are purely negative feedback loops (NFBL, oscillatory type): *rob*↔*marA* a SCC by itself; *soxS*↔*fur* in the *soxS* SCC; and *gadX*↔*hns* and *cspA*↔*hns* in the *pH* SCC. All of these are autoregulated, but autoregulation in *gadX*↔*hns* (Fig. 19.2) makes it the Smolen oscillator, a more robust type of oscillator (Stricker et al., 2008).

19.5.4 Additional electronic circuits: toggle-switch and clock-type

There are also three pairs of nodes that can exhibit various behaviors, given that they can send various types of regulation messages between them. For example, *crp* can send either an activation or repression signal to *fis*, which means that *crp-fis* can be a mutually repressed circuit (toggle-switch type) or an NFBL circuit (oscillating type), possibly even a Smolen oscillator given its autoregulations. A similar situation occurs for *fnr-arcA* in the *soxS* SCC, which can be either mutually repressed or Smolen. Lastly, *gadW-gadX* in the *pH* SCC, in which both genes send both activating and repressing signals, can be a mutually repressed, an NFBL, or a positive autoregulation (PAR) feedback loop such as in a lock-on circuit. On top of this, one of the possible NFBL configurations includes a Smolen oscillator.

There are also three FFF circuits, shown in Fig. 19.7, connecting to the *pH* SCC: one to the master regulator *crp-fis* SCC, and two others that connect through different paths to the *soxS* SCC. Remarkably, the three FFFs are regulated by the same clock: transcription factor *ihfAB*. For the three FFF circuits, the underlying feedback loop is a double-positive autoregulation (double-PAR) feedback loop, which works as a bistable lock-on circuit, which is less dynamic.

Only one *lock-on* feedback loop is present, and it is part of the FFF type circuit. Table 19.4 shows the rich variety of electronic circuits found in *E. coli*, as well as in another well-studied bacterium *B. subtilis*, as calculated in (Álvarez-García et al., 2025a).

Circuit type	<i>E. coli</i>	<i>B. subtilis</i>
Toggle-switch type	<i>galS-galR</i> , <i>uxuR-exuR</i> , <i>csgD-fлиз</i>	<i>lexa-rocr</i> , <i>glnr-tnra</i>
Oscillator type	<i>rob</i> \mapsto <i>marA</i> , <i>soxS</i> \mapsto <i>fur</i> , <i>cspA</i> \mapsto <i>hns</i> , <i>gadX</i> \mapsto <i>hns</i> ,	<i>sigK</i> \mapsto <i>gere</i> , <i>sigA</i> \mapsto <i>spo0a</i>
Lock-on types		<i>sigF-sigg</i> , <i>sigA-sigh</i> , <i>sigA-sigd</i> , <i>sigd-swra</i>
Capable of various types	<i>cpr-fis</i> , <i>gadW-gadX</i> , <i>fnr-arcA</i>	
FFF type	<i>ihf</i> \mapsto $\{fis, fлиз\}$ \mapsto <i>gadX-gadE</i> <i>ihf</i> \mapsto $\{fnr, fлиз\}$ \mapsto <i>gadX-gadE</i> <i>ihf</i> \mapsto $\{fnr, fлиз\}$ \mapsto <i>gadW-gadE</i>	

Table 19.4: **List of gene circuits in *E. coli* and *B. subtilis*.** The circuits found in *E. coli* are described in detail in Fig. 19.6. Table reproduced from (Álvarez-García et al., 2025a).

19.5.5 Structure of cycles

Cycles are very important, as we have seen before. They are a way to create longer memory between logic circuits, and correspond to different ways of regulating them.

By definition, an SCC is basically a complicated arrangement of feedback loops between its constituent nodes. As such, we want to probe its structure by studying the independent simple cycles present in the minimal TRN. Each of these cycles represents longer-term memory where signals loop around, as well as being responsible for the interconnectedness of the logic circuits.

Definition 19.3 Simple cycle. A *simple cycle* is a cycle such that every node is traversed only once, and the list of independent cycles corresponds to those that cannot be constructed from other cycles.

SCCs can be thought of as ‘information vortices’ where the signals can cycle. This is represented as the AR loop in every SCC in the effective minimal TRN in Fig. 19.1. These vortices are relevant because they are constituted from feedback loops, without which the TRN’s computational capacity would be drastically reduced to a ‘combinatorial’ nature only. Indeed, feedback loops are what allow for the more complex dynamics of the logic circuits, and this is why they are embedded in the SCCs. The SCCs, being information vortices, correspond to a very intricate cobweb of feedback loops.

To study this phenomenon, Álvarez-García et al. (2025a) look for all the independent simple directed cycles in the network. They look for a list of the independent simple cycles, the *cycle base*, since all other cycles can be constructed as a sum of these, given that they span the cycle vector space (Kavitha et al., 2009; Gruber, 2012). This is related to the

concept of Betti numbers in simplicial homology (Munkres, 2018), where the k -th Betti number is the number of k -dimensional holes in a topological space. A cycle in this sense describes a type of topological hole. In particular, for undirected graphs the first Betti number $b_0 = |CC|$, where $|CC|$ is the number of connected components, while the next Betti number $b_1 = |E| - |N| - |CC|$ is the number of independent simple cycles (Berge, 2001).

To construct this cycle base, cycles must be allowed to traverse a directed edge in the opposite direction, represented by a negative contribution. For example, in a feed-forward loop motif we can construct a cycle by traversing at least one of the edges in a ‘negative’ direction (Kavitha et al., 2009). This does not make biological sense, however, so we work with the ‘incomplete’ base of cycles allowed only by existing pathways in the network, whose combinations span only the biologically existing pathways.

Under this mathematical restriction, in directed graphs, determining the number of cycles is an NP -complete problem (Gruber, 2012). However, the networks studied are small enough to be analyzed using state-of-the-art techniques, producing output in a matter of minutes. The algorithm employed is loosely based on *Johnson’s algorithm* (Johnson, 1977). It starts by breaking the network into SCCs and searches for cycles within each. Each component is analyzed by (i) enumerating the nodes in the SCC, (ii) choosing one node (the initial/final node of the cycle), (iii) looking for the outgoing neighbors of this node, and (iv) looking for all the simple paths (paths without repeating nodes) back from each neighbor to the initial/final node.

In total, we find 41 simply directed cycles in the minimal core of the *E. coli* TRN (Fig. 19.8). Only one does not cross any logic circuit, which supports the point that connectivity between the different cycles is an important feature of these networks.

Four of them are the two-node SCCs, five are located in the *soxS* SCC, and the remaining 32 in the *ph* SCC. Each cycle contains a pair of nodes in a logic circuit, so all cycles longer than two nodes pass through two nodes that are also connected by a logic circuit. All logic circuits are, of course, two-node cycles.

For example, in the case of the *soxS* SCC we observe 2 two-node cycles (circuits *soxS* \mapsto *fur* and *fnr-arca*). The remaining three cycles in this component all pass through *soxS* \mapsto *fur*, and as such can be considered to be longer loops from *soxS* to *fur*, as in Fig. 19.8. There are numerous cases where the cycles even cross through multiple nodes that are connected by a logic circuit; this can also be seen in Fig. 19.8 where the longer loops cross through *gadE-gadW*, *gadW-gadX*, and *gadX-hns*, all of which are parts of different logic circuits. In a way, all the loops shown can be considered to be loops of various lengths between the circuits *soxS* \mapsto *fur* and *csgD-fliz*. A full list of all the cycles and code is provided at the repository <https://github.com/makselab/MinimalTRNCodes>. In this analysis we ignore loops since they connect a node to itself in a cycle of length 1, so they do not really contribute to the computational machinery.

19.6 Assembling the TRN from its building blocks

Recall the statement in (Kondev, 2014) that the most important decision for bacteria (and most humans) is ‘To eat or not to eat sugar’ (Fig. 19.9). Crucial to understanding these decision-making routines are the embedded logical gene circuits in the TRN with their respective inputs. As explained in Chapter 18, it is symmetry breaking that allows these circuits to make binary decisions—as for the toggle-switch, whose inputs break symmetry and force the circuit into one of two possible states. Besides this, it is necessary to understand how these circuits are connected and embedded in real networks, to figure out how these circuits are utilized in living cells and how their dynamics differ. This approach complements what has been done so far by the synthetic biology community, implementing such circuits *in vivo* and working with them independently of the rest of the TRN.

In Section 5.3 we discussed how to conceive of such a networked system as some form of von Neumann cellular automaton: every node can be thought of as a finite-state machine, which has a defined state at every moment, and whose future state is determined predictably by its current state and the inputs from its neighbors. We have also shown that this network is composed in part of (binary) memory devices in the form of toggle-switches and timekeeping devices, or synchronized clocks in the form of genetic circuit oscillators. Therefore this network can be interpreted as a von Neumann cellular automaton, hence as a simple type of computer. From a modern perspective any computer, at the most basic level, is composed of memory devices and timekeeping devices (Horowitz and Hill, 2015; Tanenbaum, 2016). Therefore the reduced TRN, hence the global TRN, can be regarded as a simple logical computer. Indeed, Sidney Brenner has argued that cells are good examples of Turing and von Neumann machines. This motivates the question of how these circuits are integrated into real genetic networks, and how understanding this would help to illuminate the process of decision-making by the cell, viewed as a computational process.

We can understand the bacterium as a simple logical computer: the cell takes signals from the surrounding environment as inputs, and computes a proper response as an output (a change in the expression levels of genes) which then propagates outward from the core driver network.

Ultimately, every single gene in the bacterial TRN is classified as belonging to one of the following classes (Fig. 19.10):

- $|r, l\rangle$: a synchronized symmetric fiber characterized by a building block made of either $r = n = 0, 1, 2$ loops and l ‘external regulators’, or a fractal dimension r of Fibonacci building blocks, and multilayer composites of these basic circuits.
- Combinatorial logic circuits belonging to a few strongly connected components made of broken fibration symmetries, which play the roles of digital electronic circuits for memory storage flip-flops or oscillatory clocks.
- A series of external 1-, 2-, and 3-regulators connecting the SCCs of the structure.

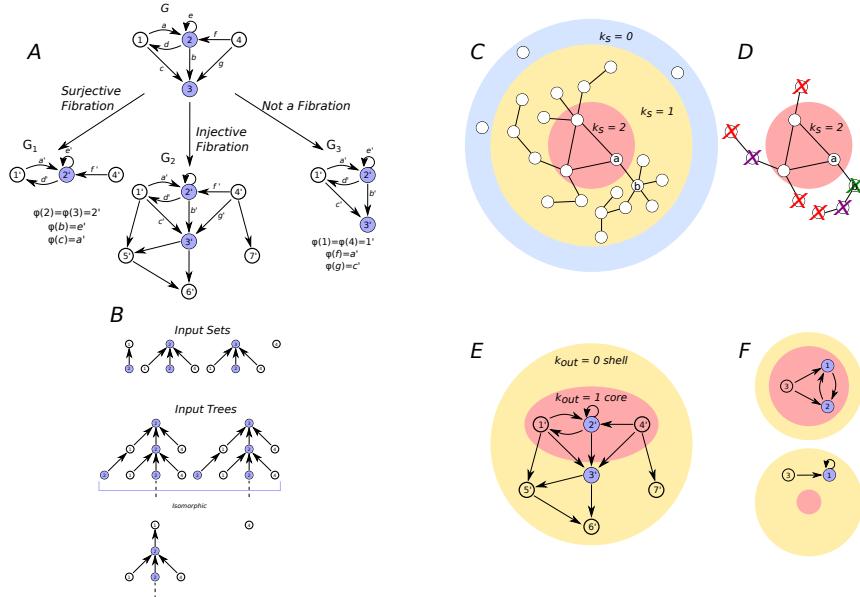
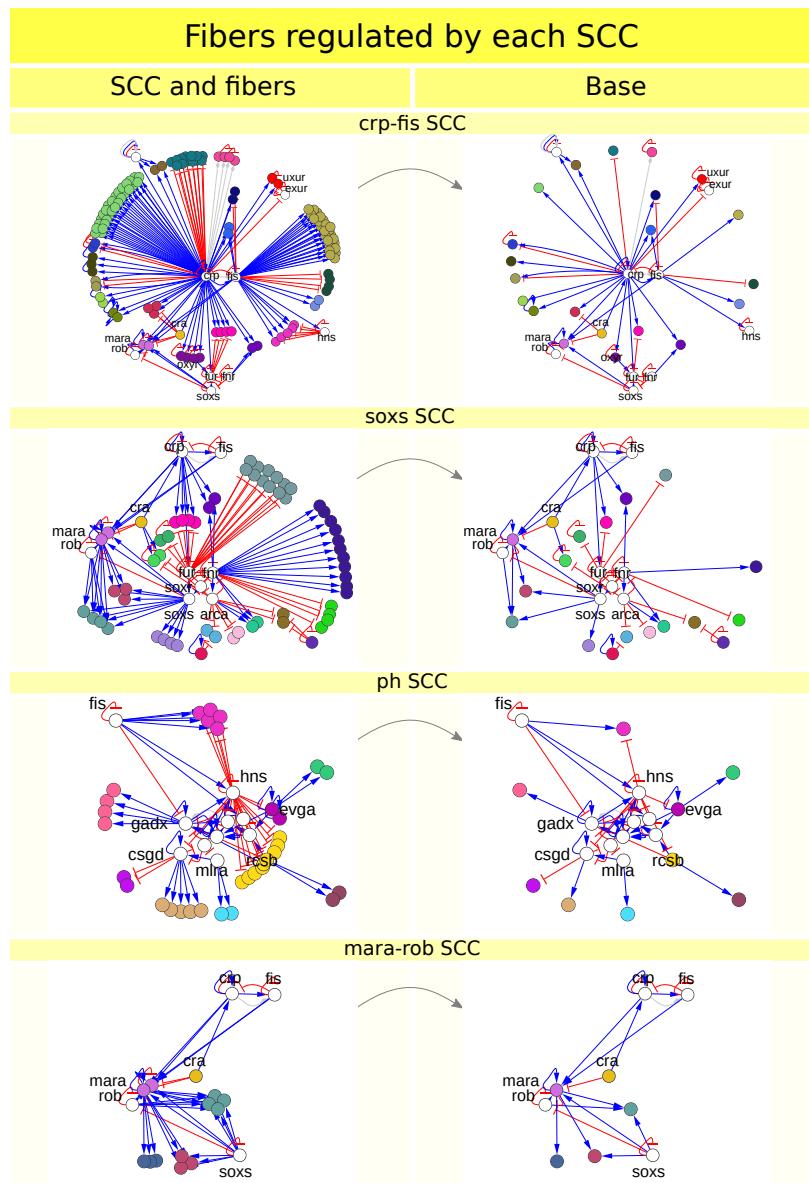


Fig. 19.3

Injective fibration and k -core decomposition. (a) Graph G , a subgraph of the TRN of *E. coli*, shows a Fibonacci building block. The map from graph G to graph G_1 , on the left, corresponds to a surjective fibration: all nodes with isomorphic input trees are collapsed to one (nodes 2 and 3 collapsed to 2'), and all input trees are preserved. The map from graph G to graph G_2 is an injective fibration: all input trees are preserved and extra nodes are included. The map from the graph G to graph G_3 , which maps node 4 to 1' does not correspond to a fibration, since the input tree of 4 (seen in (b)) is not preserved in its image node 1' in graph G_3 . The same problem occurs with the images of nodes 2 and 3 (2' and 3' respectively). (b) Input sets and input trees of nodes in graph G . (c) Schematic drawing of the k -core decomposition of an undirected network. Even though node b has a higher degree than node a , it is connected to nodes with smaller degree and has therefore a smaller coreness than node a . (d) Example of how to obtain the $k = 2$ core of the network in (c). First, the nodes with degree less than 2 are removed. The remaining nodes are those shown; successive iterations remove the nodes crossed with colored X. Different colors stand for successive iterations: red is the first iteration, the second is purple, and the last is green. (e) Example of the k -core decomposition of graph C from (a). Even though node 5' on the outer $k_{out} = 0$ shell (in red) has one output, once nodes 6' and 7' in the shell are removed, it will then be left with no output, and will be removed as well. All the remaining nodes in the $k_{out} = 1$ core have at least one output after performing this process. (f) The bottom network corresponds to the symmetry fibration of the network at the top. The SCC of nodes 1 and 2 is also a fiber, and after collapsing the fiber the SCC is lost, so the collapsed network becomes an acyclic graph, which does not have a $k_{out} = 1$ core. Figure reproduced from (Álvarez-García et al., 2025a).

**Fig. 19.4**

Symmetry fibrations applied to the main SCCs in the *E. coli* TRN. (Left) Genes in the same fiber share the same color. (Right) All fibers have been collapsed. Labeled nodes belong to SCCs or regulators to them. Blue edges with arrows represent activation, red edges with bars represent inhibition, and gray edges with rhombuses represent dual regulation (dual means that the TF can act as an activator or repressor depending on how it is activated by ligands). Figure reproduced from (Álvarez-García et al., 2025a).

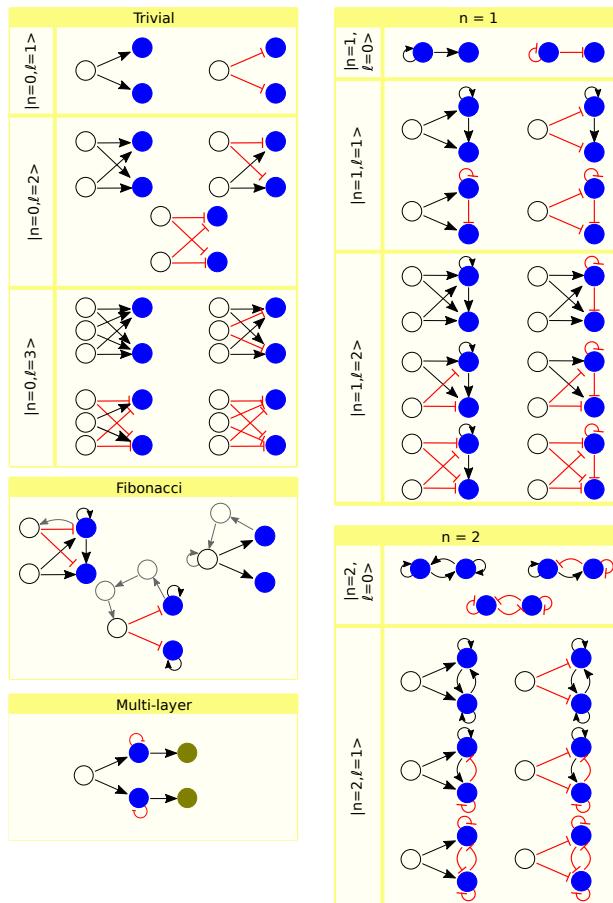
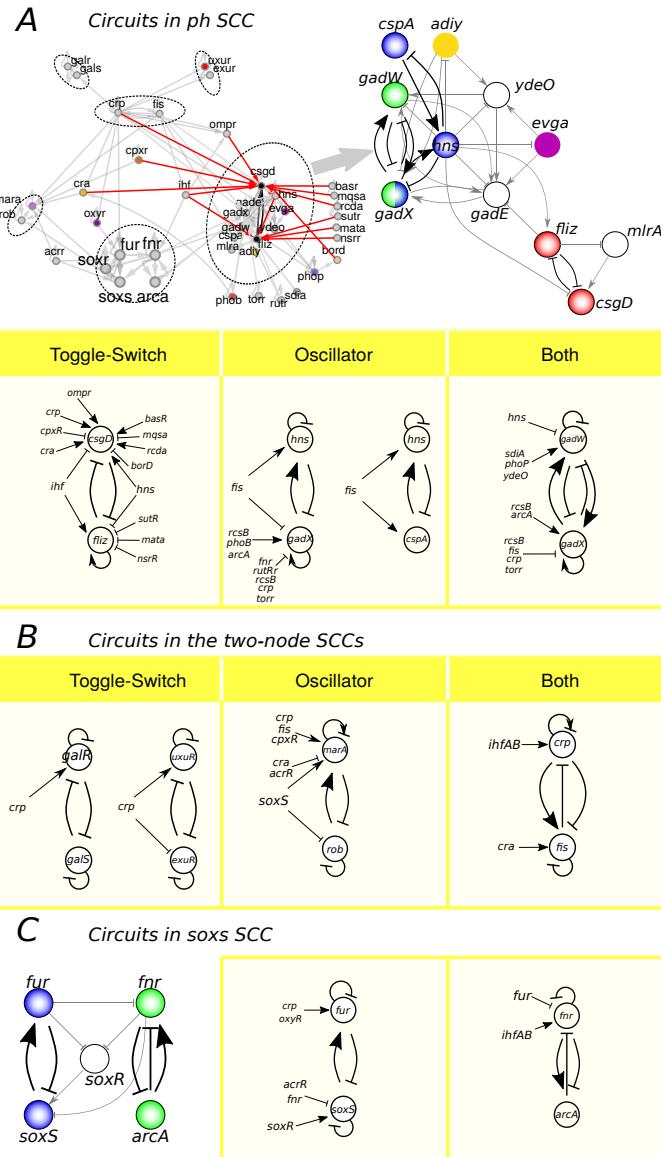
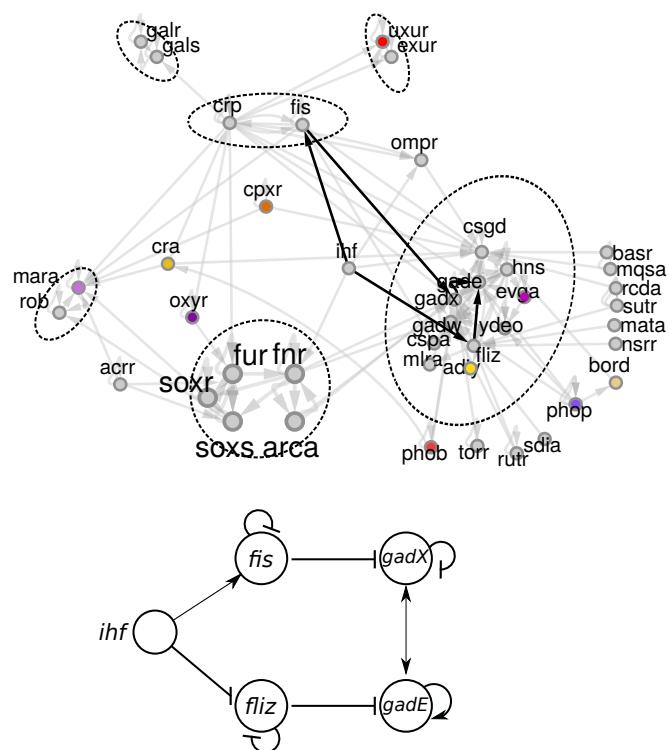


Fig. 19.5

Possible circuits of the observed building block structures from *E. coli* and *B. subtilis*. These circuits are all the possibilities for building blocks that can be used to assemble a computational TRN from the bottom up. We take the observed $|n, l\rangle$ classes in both bacteria, and show all the possible combinations of activation and inhibition regulations for each class, i.e. all possible configurations that would still have a symmetric pair of nodes. Figure reproduced from (Álvarez-García et al., 2025a).

**Fig. 19.6**

Electronic circuits found in *E. coli* TRN. (a) The largest SCC in charge of mostly pH responses, and the electronic circuits embedded in it. *E. coli*'s minimal TRN is shown with red links for the symmetry breaking inputs to the toggle-switch *fliZ*-*csgd*. (b) Electronic circuits forming two-node SCCs. (c) Electronic circuits in the *soxs* SCC. Nodes involved in toggle-switches are colored in red in the SCC depictions, oscillating nodes in blue, and nodes in circuits that can be either in green. For every circuit, the incoming signals that break the symmetry are shown. Figure reproduced from (Álvarez-García et al., 2025a).

**Fig. 19.7**

FFF electronic circuits. *Top:* These circuits connect the main SCC *crp-fis* to the pH-SCC. *Bottom:* The isolated circuit. Figure reproduced from (Álvarez-García et al., 2025a).

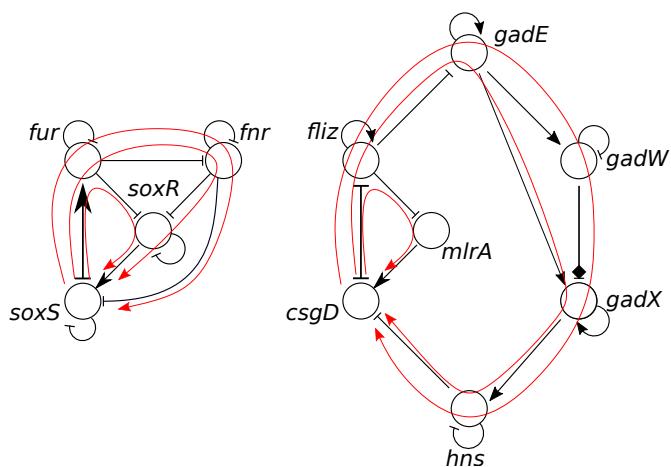


Fig. 19.8

Example of simple directed cycles in *E. coli*. All are different cycles that cross through the logic circuits *soxS-fur* on top and *csgD-fliz* on the bottom. Figure reproduced from (Álvarez-García et al., 2025a).

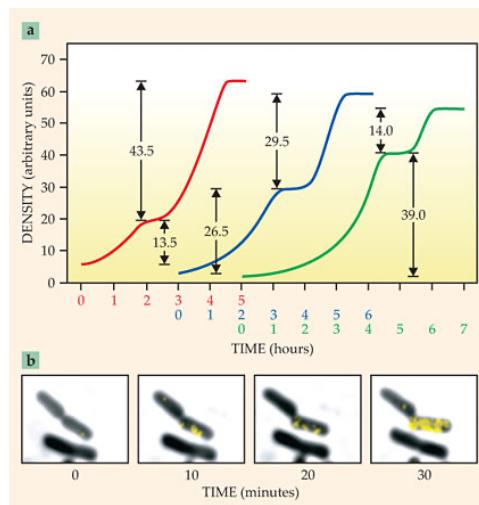


Fig. 19.9

Bacterial decision making in action. (a) Growth of *E. coli* bacterial culture in an environment of glucose and lactose mixed in proportions 1:3, 1:1, and 3:1 (curves from left to right, respectively). The observed amount of growth in each phase (see the plateau) is proportional to the amount of glucose and lactose in the cell's environment. Figure reproduced from (Müller-Hill, 1996). (b) In an environment with lactose, cells switch to a cellular state that can digest lactose as indicated by the appearance of yellow-labeled lactose permease from left to right. Figure reproduced from (Choi et al., 2008). The bacterial cell controls these states like a computer' using the flip-flop formed by the *cpr-fis* genes. They are the master regulators of the sugar utilization circuits catabolizing the different sugars. Figure reproduced from (Konddev, 2014) with the permission of AIP Publishing.

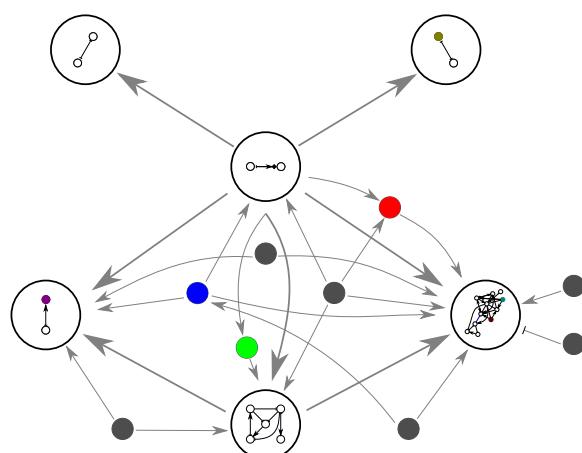


Fig. 19.10

Effective representation of the minimal TRN of *E. coli*. The structure is feed-forward with the carbon SCC at the center, with autoregulation loops inside the SCCs and single gene regulators that regulate 1, 2 and 3 SCCs.

From Structure to Function: Cluster Synchronization in Genetic Networks

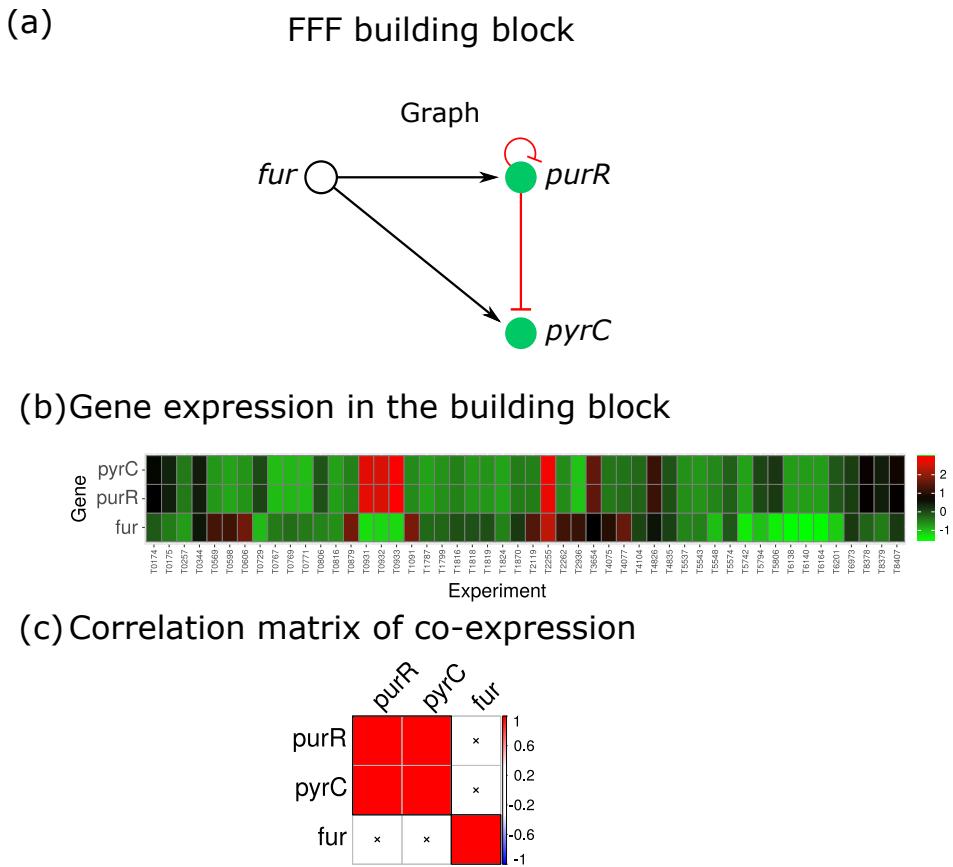
What does the existence of fibrations mean from a biological standpoint? In genetics, fibration synchrony means coexpression. In this chapter we consider experimental expression profiles predicted from gene fibers. We show that gene expression levels of the genes inside each fiber synchronize in the manner predicted by fibration theory. Genetic synchronization provides the functional relation, and symmetry is the link between the structure of the underlying TRN and its function via synchronization of its constitutive units. This closes the gap from structure to function.

20.1 Structure ↽ function in genetic networks

Systems biology has developed the capacity to look at the activity of the entire set of proteins of an organism, such as the 100,000+ proteins expressed by the ~20,000 human genes. This expression can be monitored simultaneously for all genes using transcriptomic techniques like microarrays or RNA-seq. When the patterns of gene expression are observed through correlation functions, it is found that the genes separate into groups. These groups are sets of genes with a similar activity or gene coexpression, in a statistical sense, implying cluster synchronization (Klipp et al., 2016; Aguiar et al., 2022).

The subject of this chapter is whether this form of synchronization can be predicted from fibrations in the underlying TRN. Gene fibrations provide a way to ensure gene coexpression by a specific network wiring, without necessarily placing the genes in the same transcription unit (i.e., under the control of the same promoter), and even without necessarily placing the genes in the same regulon under the control of the same transcription factor.

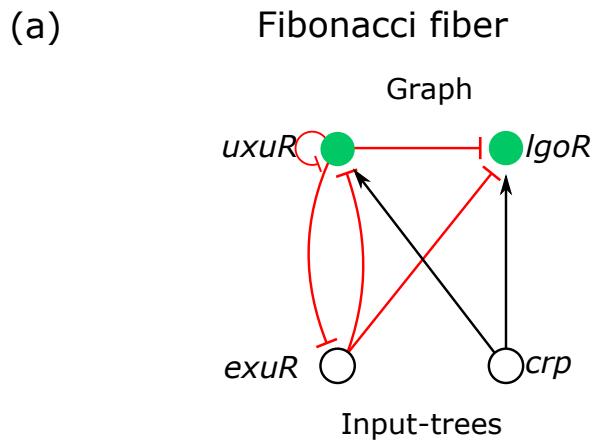
In general, we seek groups of genes that have similar gene expression activity since these similarities are examples of cluster synchronization, revealing common biological functions. Gene annotation based on patterns of gene expression have been widely used in the literature (Klipp et al., 2016). Typically, a heat map measures the expression of genes under different conditions, such as those in Figs. 20.1 and 20.2 for *E. coli*. These heat maps are obtained from databases that compile a large number of experimental conditions over many laboratories in the world and over many species. In a heat map, genes are rows, while different experiments, conditions, or subjects appear in columns. Clusters across the rows produce sets of genes with highly correlated gene expression, which can be interpreted as working in synchrony, see (20.2) below, across different experiments, and therefore potentially suggest similar functional annotations for the genes. Clustering across the columns, instead, leads to sets of similar conditions.

**Fig. 20.1**

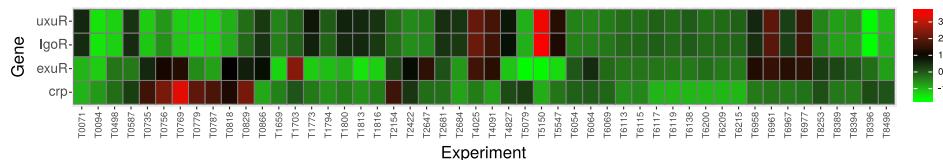
Heat map and correlation matrix of an FFF in *E. coli*. Genes are rows, and experiments are columns. The color bar indicates the differential expression level in AU with respect to the wild-type condition of a given gene and experiment.

Remark In a heat map, genes cluster over conditions/subjects, which contrasts with the definition of synchronization as the similarity of gene expression of a single condition over time. It is possible to identify the correlation over conditions/subjects with the synchrony over time if we assume ergodicity in the data. We elaborate on this in Section 20.3, see Eqs. (20.1) and (20.2).

Under this assumption, we can study correlation profiles of gene expression in search of synchronized genes, and then test whether this notion of synchrony emerges from the fibers in the underlying TRN, bridging the gap between function and structure. We test these ideas with transcriptome data. Based on idealized ODE models of gene expression, we discuss whether the predicted synchronization in expression profiles for genes in fibers can, in reality be measured from the experimental correlated expression profiles where the idealized symmetry is only approximate.



(b) Gene expression in the building block



(c) Correlation matrix of co-expression

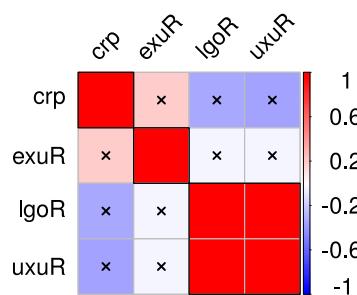


Fig. 20.2

Heat map and correlation matrix of a Fibonacci fiber in *E. coli*. Genes are rows and experiments, are columns. The color bar indicates the differential expression level in AU with respect to the wild-type condition of a given gene and experiment.

20.2 Structural network and functional network

Relating the structural network to the functional one is a longstanding question for all biological systems, from genes to the brain (Park and Friston, 2013; Friston, 2011; Hartwell et al., 1999a; Tononi et al., 1994).

As claimed in Section 11.3, symmetry is the missing link to uncover how structure determines function in biological networks, from the TRN to the brain. Chapters 23, 24, and 25 deal with structure-function in the brain (Park and Friston, 2013; Friston, 2011), where we consider structural networks made from axons and bundles of axons of white-matter fiber tracts. In this chapter, we distinguish between the structural network and functional networks in gene regulatory networks.

The structural network or structural graph is the TRN: the set of genes and edges that form the underlying network of physical binding between the TF expressed by the source gene and the DNA binding promoter site of the target gene.

When the structural graph is equipped with state variables and dynamical equations, it becomes a network ODE. When the ODE represents a lumped model of gene activity such as those studied in Chapter 8, the state variables are the concentration of proteins expressed by the gene, and the dynamical equations are those described in the lumped models used in Chapter 8.5. We are interested in cluster synchronization of these state variables.

The functional network is obtained from the correlations between gene activity. A link in the functional networks implies some common functionality between the genes. This functionality can be measured by many means; the most common is to calculate a Pearson correlation coefficient between the activity of the genes and to consider a link to exist when the correlation is above a certain threshold. Once the functional network is defined, clustering techniques can be applied to find clusters of genes. These are identified as genes of cluster synchronization.

Although the function is usually given a broader meaning, here, we identify ‘function’ with synchronization of genetic activity as measured by any transcriptomic technique.

The ‘structure \rightsquigarrow function’ concept is applied throughout the book to other examples, such as brain synchronization. This turns out to be an important concept since, somehow, the role of the functional network is sometimes confused with that of the structural network. In these examples, we clearly see that the functional network is expected to be a complete subgraph for genes that are synchronized and belong to a common fiber. In contrast, the structure network of the fiber does not need to be fully connected, and indeed, never is. A fully connected functional network implies that all the genes can synchronize, but this does not mean that the structural network must reflect the same fully connected nature. In fact, this is never the case. Thus, structural connectivity cannot be directly inferred from functional connectivity despite many efforts in the field to prove the opposite.

20.3 Cluster synchronization from gene coexpression

Examples of synchronization are abundant in biology, where the continuous expression of genes is routinely measured. Gene coexpression indicates that proteins act as a synchronous unit in time. This represents a direct measure of synchronization via the expression activity of each gene as a function of time:

$$\lim_{t \rightarrow \infty} (x_i(t) - x_j(t)) = 0, \quad (20.1)$$

where $x_i(t)$ and $x_j(t)$ are the expression levels of genes i and j in the synchrony cluster and the equation holds for all pairs i, j in the cluster.

This direct measure of synchronization contrasts with what is usually measured in the cell by gene coexpression using transcriptomic techniques such as microarray and RNA-seq experiments (Barrett et al., 2012). These are indirect measures of synchronization, because gene activity is not measured as a function of time but is measured across different experimental conditions or across subjects of a given organism. These gene expression profiles are snapshots of the expression levels (usually mRNA concentration) of all the genes measured at once and then repeated over experimental conditions or different subjects. In this case, the time series of (20.1) are replaced by T data points of gene activity $x_i(t_k)$ with $k = 1, \dots, T$, over different experimental conditions and/or subjects, creating a discrete series of T events. Perfect cluster 'synchrony' over all conditions then implies:

$$(x_i(t_k) - x_j(t_k)) = 0, \quad \forall k = 1, \dots, T. \quad (20.2)$$

Clearly, the two conditions, (20.1) and (20.2), need not be the same. Thus, replacing the condition for synchronization over time in Eq. (20.1) with a synchronization over conditions/subjects in Eq. (20.2) is not always possible and should be considered an approximation. When these conditions are the same, the system is said to be ergodic. Our assumption in the rest of the book is that the synchronization over experimental conditions is a good proxy of time synchronization.

Synchronization is never perfect, as expected from (20.2). Instead, it occurs in real systems as a degree of synchronization in the data that can be captured by statistical techniques.

The most common and simpler (linear) statistical method is the correlation function, which defines a correlation matrix C_{ij} from the Pearson correlation coefficient. It quantifies synchrony between genes i and j over T conditions as:

$$C_{ij} = \frac{1}{T} \sum_{k=1}^T \left(\frac{x_i(t_k) - \mu_i}{\sigma_i} \right) \left(\frac{x_j(t_k) - \mu_j}{\sigma_j} \right), \quad (20.3)$$

where $x_i(t_k)$ is the mRNA concentration of gene i in condition t_k , and μ_i and σ_i are the mean and standard deviation of $x_i(t_k)$ averaged over all conditions:

$$\begin{aligned} \mu_i &= \langle x_i(t_k) \rangle \\ \sigma_i &= \sqrt{\langle x_i(t_k)^2 \rangle - \langle x_i(t_k) \rangle^2}. \end{aligned} \quad (20.4)$$

Here, $\langle \cdot \rangle$ represents a sample average taken over the series of different conditions (or over different individuals): $\langle x_i(t_k) \rangle = (1/T) \sum_{k=1}^T x_i(t_k)$, when T measurements are taken, each one at time t_k . Here, t_k is just an index and nothing depends on time.

The correlation C_{ij} ranges from -1 to 1 . $C_{ij} > 0$ corresponds to positive correlation between genes i and j , $C_{ij} < 0$ corresponds to anticorrelation, and $C_{ij} = 0$ indicates lack of correlation.

The correlation matrix can be calculated not only from Pearson correlation coefficients like (20.3), but from many other measures of synchrony. In fact, there is a zoo of strategies to interrogate synchrony between any given dynamical variables, not only for gene expression but also for any other biological time series, such as neural activity. The most commonly used and simplest measure is the linear correlation function of (20.3) and covariance.

However, this correlation measure may not capture the correct synchrony in the signal. For instance, two signals may have different amplitudes yet be synchronous in phase. In some applications, it is desirable to capture this kind of synchronization, particularly in the brain. The Phase Locking Value (Bruña et al., 2018; Lachaux et al., 1999) captures this kind of synchronization, which is relevant, for instance, for synchronization in the brain. We discuss this in Chapters 23 and 25, where we also employ a more direct measure of synchrony, the *Level of Synchronicity* (LoS), which directly measures (20.1), and provides a stronger condition of synchrony. To be considered ‘fully’ synchronous, the neurons need not only be active at the same time (phase synchrony), but also to have the same value (amplitude synchrony).

There are also nonlinear measures, such as mutual information and covariograms, that focus on the statistical dependence of two signals. Maximum entropy methods and Bayesian networks are also very popular ways to obtain effective models and functional networks from data.

Together, these methods let us derive a correlation matrix from the data. Once the correlation matrix C_{ij} is known, different methods can be used to obtain synchrony clusters from it. In general there are two methods:

- Threshold this matrix to obtain the functional network, which is then used to obtain the synchronous clusters by network analysis (Section 20.3.1).
- Analyze C_{ij} directly by any hierarchical clustering algorithm (Section 20.3.2) to obtain the synchrony clusters.

20.3.1 Thresholding method to find cluster synchronization

The simplest way to obtain a functional network is by thresholding the correlation function. This creates undirected links between nodes i and j if the correlation exceeds a given threshold value p (Bullmore and Sporns, 2009; Rubinov and Sporns, 2010; Gallos et al., 2012b). Thresholding, in principle, produces the adjacency matrix of the functional network A_{ij} by including an undirected edge when the correlation between i and j is higher than the threshold p . However, C_{ij} can be positive, signaling correlations, or negative, signaling anticooperations. This introduces a dilemma that has to be solved for the particular analysis. In general, correlation and anticooperation are two types of synchrony. They are two extreme

cases in a continuum of different types of oscillating data, which arise from a phase shift that can take any value between zero (correlation) to 180° (anticorrelation). Unfortunately, (20.3) ignores all these intermediate values, while the phase locking value captures this more refined level of synchrony.

Different choices of the threshold p can produce different adjacency matrices, leading to different types of functional networks. They can be unweighted ($A_{ij} = \{0, 1\}$), signaling the absence or presence of an edge, yet neglecting anticorrelations:

$$A_{ij} = \begin{cases} 1 & \text{if } C_{ij} > p, \\ 0 & \text{otherwise.} \end{cases} \quad (20.5)$$

Alternatively, we can consider not only correlations but also anticorrelations, as proxies of synchrony:

$$A_{ij} = \begin{cases} 1 & \text{if } |C_{ij}| > p, \\ 0 & \text{otherwise.} \end{cases} \quad (20.6)$$

While this last definition includes anticorrelations, it places them on the same footing as positive correlations. That is, the method does not distinguish between both, which is in general, not advisable.

If we assign the weight information to the edges, we obtain a weighted functional network ($A_{ij} \in \mathbb{R}$) by considering the strength of the correlation interaction. Here there are also two alternatives. We can ignore negative correlations:

$$A_{ij} = \begin{cases} C_{ij} & \text{if } C_{ij} > p \\ 0 & \text{otherwise.} \end{cases} \quad (20.7)$$

or we can consider anticorrelations also to be a form of synchronization:

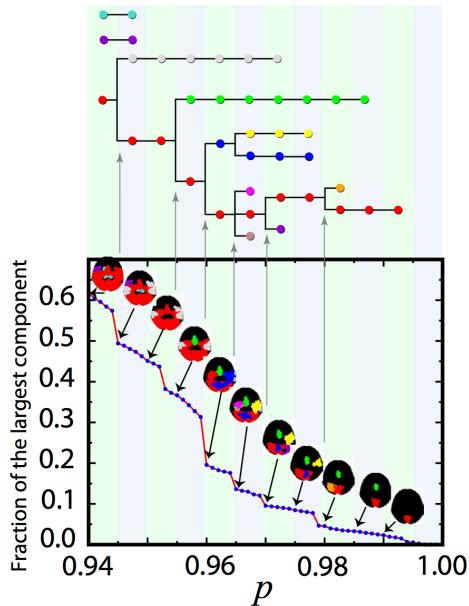
$$A_{ij} = \begin{cases} C_{ij} & \text{if } |C_{ij}| > p, \\ 0 & \text{otherwise.} \end{cases} \quad (20.8)$$

Below, we use (20.5) to analyze gene expression data, being the simplest method of all, but more sophisticated analysis can be done with more sensitive methods.

The main question is how to choose the threshold p . A large p reveals clusters of strongly correlated genes, which in principle, should be disconnected from each other. As we lower the value of p , these clusters start to connect (percolate) to each other through weak links. The choice of the threshold p is an art in itself. It appears in any analysis of clustering in diverse ways.

In principle, we could pick a unique threshold. This leads to a well defined functional network from where the clusters can be obtained by modularity, such as the Louvain algorithm (Blondel et al., 2008) of Section 17.4. Section 20.4.2 employs this simple technique to analyze gene expression in *E. coli*.

There are more elaborate ways to pick the threshold, which can reveal more structure in the data. Gallos et al. (2012b) and Reis et al. (2014) demonstrate that by applying percolation analysis to C_{ij} , a hierarchical structure emerges with critical values p_c that reveals modules of cluster synchrony. Such an analysis has been applied to fMRI brain signals in (Gallos et al., 2012b; Gili et al., 2025) and is discussed in more detail in Chapter 25.

**Fig. 20.3**

Hierarchical formation of synchrony clusters in a brain network. An example of a brain functional network from human fMRI in a dual task experiment from (Sigman and Dehaene, 2008), as analyzed in (Gallos et al., 2012b). The size of the largest connected component of nodes (as measured by the fraction of the total system size) is a function of the percolation threshold p . As we lower p , the size of the largest component increases at jumps when new modules emerge, grow, and finally are absorbed by the largest component. We display the resulting clusters for the indicated p . The jumps make the identification of clusters clear. The tree in the upper figure shows the hierarchical merging of clusters into a hierarchical network of networks. The topography of these clusters reflects coherent patterns of cluster synchrony for a visual/auditory activity. We observe clusters covering the anterior cingulate (AC) region, a cluster covering the medial part of the posterior parietal cortex (PPC), and a cluster covering the medial part of the posterior occipital cortex (area V1/V2) along the calcarine fissure.

We monitor the size of the largest connected component of the functional network as we lower p from 1 to 0, as in Fig. 20.3. This is interpreted as a percolation process (Bunde and Havlin, 1999). The jump in the largest connected component at p_c signals the merging of two clusters, A and B, formed by strong links for $p > p_c$ and interconnected by weak links with $p < p_c$. The clusters A and B are composed of nodes with incoming links inside their clusters, defined by their strong links. Each node in A or B has a number of outgoing weak links connecting to nodes in the other clusters. Within each cluster, the nodes are synchronous. Another question is whether these clusters are also synchronous between themselves. We investigate this question in Chapter 25.

Finer modules can be obtained by lowering the threshold. Figure 20.3 shows the detailed behavior of a modular network obtained from an fMRI BOLD signal in an individual performing a dual visual and auditory task obtained in (Sigman and Dehaene, 2008) and analyzed in (Gallos et al., 2012b). The hallmark of hierarchical cluster synchronization is the jumps in the percolation process seen in Fig. 20.3. The figure shows the size of the largest connected component during percolation for the correlation function as we lower a threshold p . At larger values than $p = 0.98$, three large clusters are formed localized to the medial occipital cortex (red), the lateral occipital cortex (orange), and the anterior cingulate (green). Critical values occur at the jumps. At $p_c = 0.978$ the orange and red clusters merge, as revealed by the first jump in the percolation plot. As p continues to decrease, this mechanism of network formation and absorption repeats, defining a hierarchical process depicted in the tree in the top panel of Fig. 20.3.

This mechanism represents the iteration of a process by which clusters form at a given p value and merged via comparably weaker links. This process is recursive. The weak links of a given transition become the strong links of the next transition, in a hierarchical fashion, resulting in a hierarchical percolating process where clusters merge with each other, and sharp transitions are observed in the size of the largest network (red in Fig. 20.3) as it absorbs smaller ones. The modules have consistent topographic projections on the map of the brain (Gallos et al., 2012b,a). The signature of hierarchical synchronization in a brain structure is, then, the hierarchical sequence of clusters as a function of the threshold p , as shown by the jumps in Fig. 20.3. Bardella et al. (2016) applied a modified percolation method, where rather than monitoring the jumps in the largest connected component, the number of components is followed.

This percolation analysis is not restricted to the brain but can be performed on any data that is expected to cluster; in particular, gene coexpression data. In Chapter 25 it will be applied to gene coexpression patterns in the human brain. This chapter also develops a more sophisticated thresholding method, called clique synchronization (23.4), which captures synchrony within the data better than the simple percolation thresholding of C_{ij} does.

Thresholding C_{ij} is a way to regularize the data, and this is the domain of machine learning. A widely used thresholding method in machine learning is the Graphical Lasso (Glasso). There are countless methods in machine learning to obtain sparse representations of data, such as Ridge regression or Lasso (least absolute shrinkage and selection operator), a regularization technique introduced by Tibshirani (1996). Here we elaborate on Glasso (implementation available at <http://www-stat.stanford.edu/~tibs/glasso>), which uses Lasso to obtain a sparse representation of the data and then provide a graphical representation by thresholding spurious correlations. This is completely analogous to thresholding C_{ij} (Sojoudi, 2016).

To avoid spurious correlations introduced by random covariates, and to reduce the dimensionality of the problem, graphical lasso (Friedman et al., 2008) infers a sparse representation \hat{J} from the correlation matrix C_{ij} . Graphical lasso assumes a multivariate Gaussian distribution for the dynamical variables $x_i(t)$, and then infers the model's parameters from experimental data. The estimation of this matrix is made by minimizing the log-likelihood

of a multivariate normal distribution:

$$\log \mathcal{L}(\hat{J}) = \log \det[\hat{J}] - \text{Tr}[\hat{C}\hat{J}] - \lambda|\hat{J}|, \quad (20.9)$$

where the penalty parameter λ controls the sparsity of the resulting network as estimated by the number of total connections. Here λ plays the same role that p does in the thresholding method (Sojoudi, 2016). After obtaining \hat{J} for a given λ , a graph of the functional network is obtained by applying a small threshold ε to \hat{J} , such that spurious interactions are avoided. That is, an edge in the functional network exists if $\hat{J}_{ij} > \varepsilon$. This produces a sparse representation of the functional network.

As in the thresholding method, the critical step is to choose λ . To select the appropriate value, two criteria can be applied, which are valid across all biological data: they are clustered, yet integrated (Tononi et al., 1994). This means that the existence of clusters does not imply that these clusters are completely isolated. They must still be integrated into the system as a whole. In neuroscience, this is referred to as the ‘integration versus segregation problem’, and it is a longstanding issue to understand this problem for the brain (Tononi et al., 1994). Brain networks are integrated, i.e., signals are exchanged through the whole architecture within and between different brain clusters. Therefore, the functional network must guarantee global connectivity. At the same time, the functional brain network is segregated into clusters.

Based on these considerations, the value of λ was fixed by Del Ferraro et al. (2018) so that all nodes are connected through a path (to allow for brain integration) and, at the same time, are segregated into clusters. Their procedure follows a percolation process similar to thresholding the correlation matrix, as in Fig. 20.3. We decrease the value of λ to regularize the functional network less and less until all relevant clusters are integrated into a global network, meaning that they are all connected. For this, we monitor the jumps in the size of the largest connected component of the functional network to identify the weak and strong links defining the clusters.

What is interesting is that it has been mathematically proved by Sojoudi (2016) that the functional network obtained by thresholding C_{ij} is exactly the same as the one obtained by the Glasso J_{ij} . That is, if we build a functional network by thresholding C_{ij} by p and a network by Glasso with penalty λ , the networks are exactly the same when $p = \lambda$. This remarkable result brings some sort of validity to the whole method. It also facilitates obtaining a sparse representation of data by just thresholding C_{ij} , which is a much faster operation than inverting Glasso to obtain J_{ij} . In fact, Glasso cannot be applied to very large network sizes, while thresholding can.

After the functional network is obtained, either by thresholding C_{ij} or by Glasso or any other method, the clusters of synchrony can be obtained by applying any clustering method to the resulting graph. The most common method is to apply modularity or community detection algorithms (like Louvain, Section 17.4). These can identify clusters that are highly connected internally, but have few connections to other clusters. Be reminded: modularity algorithms should be applied to the functional network, but not to the structural one.

Modularity produces modules that are just an approximation to cluster synchrony. In principle, a perfectly synchronous cluster is a clique of nodes, fully connected among themselves but with no connections outside the cluster. However, this ideal case never

occurs in nature, so modularity is just a first approximation used to find them in real data. In Section 23.5.5 we discuss a more refined method based on clique synchronization algorithm that better captures these clusters.

20.3.2 Hierarchical clustering to find cluster synchronization

There are also alternatives to obtain clusters directly from the correlation matrix C_{ij} without going through the functional network. Clustering is a well-developed branch of machine learning, and a large number of techniques exist.

Unsupervised machine learning clustering methods are hierarchical clustering and agglomerative and divisive class, determined by Euclidean distance and others. Some of these methods have been discussed in Section 17.4. In hierarchical modularity (Fig. 17.6), the dendrogram can be cut at any level to partition the data into clusters. Again, this permits a degree of liberty to choose the thresholding level, and again, this parameter should be determined in advance. In principle, thresholding the dendrogram should be analogous to thresholding C_{ij} , although the resulting clusters are not the same.

Other unsupervised methods include K -means, which requires specifying the number of clusters in advance to find the set of clusters that minimizes the distance from each data point to the clusters. The drawback of this method is that there is no a priori information on how many clusters there are in the data; also, there is no information on the centroids of the clusters, so random initialization of the centroids is needed. Thus, there is no guarantee of convergence with the right answer. Many other approaches improve on these methods (fuzzy k -means, etc.).

20.4 Gene coexpression synchronization analysis from massive datasets

Many specialized databases contain information on the transcriptome across species, such as Colombos (Moretto et al., 2016), Ecomics (Kim et al., 2016), and SubtiWiki (Zhu and Stüleke, 2018). On another level, there are databases of gene annotation that are manually curated to reflect the interplay between different genes in their respective processes and functions. Ecomics contains microarray and RNA-seq experiments gathered from NCBI GEO (Barrett et al., 2012), including several *E. coli* strains in 3,579 experimental growth conditions for 4,096 genes. The SubtiWiki dataset contains experiments in the GEO database that give 104 experimental conditions for *Bacillus* genes. Raw data on gene expression among different experiments and platforms is pre-processed to obtain normalized expression levels by using noise reduction and bias correction normalized data across different platforms.

Leifer et al. (2021) use these datasets to test gene synchronization via fibrations in *E. coli* and *B. subtilis*. The Ecomics portal (Kim et al., 2016) collects microarray and RNA-seq experiments from different sources including the NCBI Gene Expression Omnibus (GEO) public database (Barrett et al., 2012) and ArrayExpress (Kolesnikov et al., 2015). The data is also compiled at the Colombos web portal (Moretto et al., 2016). We use

Economics since it provides the data in wild-type (WT) conditions, rather by providing data based on the fold-change that compares a mutation or perturbation to the WT. Using Economics, we obtain a set of experimental conditions where the particular genes in a given fiber have been significantly expressed. For this task we follow standard gene expression analysis, as developed in *colombos.net* and (Moretto et al., 2016) for expression levels in *E. coli*: first to identify the set of growth conditions where the genes in a given fiber are significantly expressed with respect to random noise, and then to test synchronization through correlations in gene activity using these conditions. We then repeat the scheme using the conditions in Subtiwiki for *B. subtilis* (Zhu and Stülke, 2018).

For a given set of genes in a fiber, we find the experimental conditions for which the genes have been significantly expressed by comparing the expression samples over different biological conditions. To filter conditions where the genes are expressed we use the Inverse Coefficient of Variation (ICV), similar to that applied by Colombos.

The experimental conditions are ranked by:

$$\text{ICV}_i = |\mu_i|/\sigma_i, \quad (20.10)$$

where μ_i is the average expression level of the genes in the condition i and σ_i is the standard deviation. Moretto et al. (2016) select those conditions with $\text{ICV}_i > 1$, i.e., where the average expression levels in the particular condition i are higher than the standard deviation. This score reflects that, in a relevant condition, the genes show an increment of their expression level, above the individual variations caused by random noise. The expression profiles obtained are organized by the experimental conditions, which are labeled according to the GEO database (Barrett et al., 2012). A heat map such as shown in Fig. 20.2 is obtained. For each set of genes, there is a particular set of conditions where they have been activated, as determined by the ICV. The conditions obtained depend on the fiber and, generally speaking, any pair of fibers may have correlations calculated under suitable conditions. To deal with this issue, the union of conditions in which both fibers are activated is considered when determining correlations between nodes in different fibers. From these data, we calculate the coexpression matrix using the Pearson correlation coefficient between the expression levels of two genes under the relevant conditions. Then the functional network is obtained by thresholding, and the synchrony clusters are obtained by modularity.

20.4.1 Gene synchrony in fibration circuits

Figs. 20.1 and 20.2 show the coexpression analysis in the FFF and Fibonacci circuits in *E. coli*. The expression profiles of the genes in these circuits are filtered by ICV over all experimental conditions in Economics to obtain the heat maps. Then the Pearson correlation matrix is calculated over these conditions, as shown.

The expression profiles for the FFF fiber are composed of the main TF regulator *fur* and the fiber genes *purR* and *pyrC*, which should be synchronized. The fiber predicts no synchronization between *purR* and *pyrC* and the regulator *fur*. In the FFF, *fur* should neither synchronize with *purR* nor with *pyrC*. This is corroborated by the correlation matrix in Fig. 20.1c.

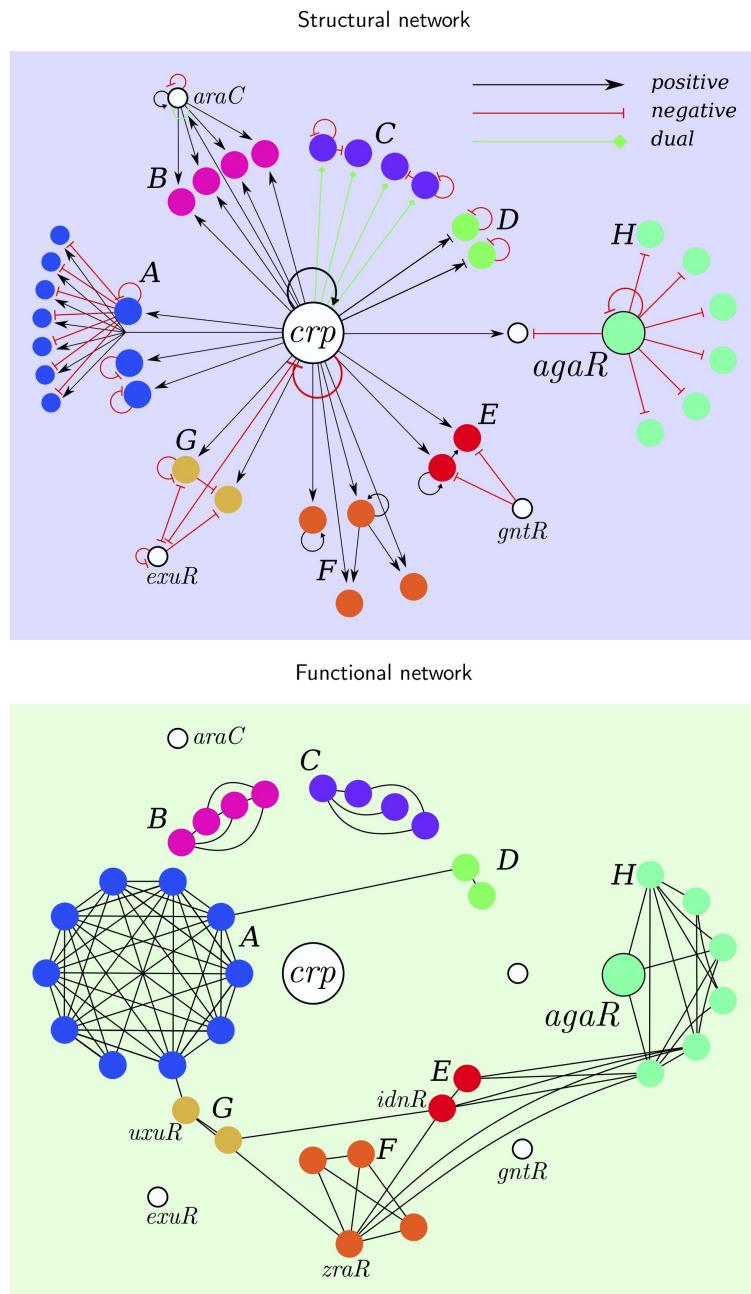


Fig. 20.4

Structure \rightsquigarrow function in the carbon utilization network of *E. coli*. *Top:* The structural network represents the topology of the gene regulatory network of alternative carbon sources. *Bottom:* Functional network obtained by thresholding the correlation matrix $C(i, j)$ in Fig. 20.5 at 0.6. That is, any two genes with correlation above 0.6 are connected, and any two genes with correlation below 0.6 are disconnected.

The lack of synchronization between the fiber genes *purR-pyrC* and its regulator occurs even though the fiber is directly regulated by *fur*, that is, direct regulation does not necessarily lead to synchronization. This is because, beyond this direct regulation, other TF regulations inside the fiber synchronize the genes within the fiber but not outside. As predicted, genes are highly coexpressed within fibers but not significantly correlated with the regulator genes.

This is corroborated by the analysis of the FF circuit in Fig. 20.2. Figure 20.2c shows strong correlations between the genes *lgoR* and *uxuR* within the Fibonacci fiber, yet the regulators *exuR* and *crp* (which also regulate many other fibers) remain unsynchronized with the fiber.

A functional network obtained by thresholding the correlation matrix is shown in Fig. 20.4, bottom. We use a threshold $C_{ij} > 0.8$ to produce a functional network where a link exists if the Pearson correlation coefficient between gene i and j , C_{ij} is above the threshold. Links in this functional network exemplify significant synchronization, with possible functional relations between genes in the fiber indicated by significant levels of synchronization.

20.4.2 Structure \rightsquigarrow function in the carbon utilization circuit of *E. coli*

Leifer et al. (2021) perform a large-scale analysis of gene synchronization expression patterns in the carbon utilization subgraph of *E. coli* controlled by the *crp* gene shown in Fig. 14.3. This subgraph controls genes involved in the metabolism of alternative carbon sources, i.e., all sugars except glucose. It contains genes involved in pathways used to catabolize sugars like fucose, galactosamine, ribose, N-acetylglucosamine, maltose, ribose, galacturonate and L-idonate, which are used as an alternative to glucose, the primary source of sugar intake. The resulting structural network of the TRN can be seen in Fig. 20.4a. We apply the fiber finding algorithm and obtain the colored fibers shown in the figure. This subgraph is part of the carbon SCC in Fig. 14.3 and analyzed in detail in Section 14.3.

Although the genes have a related general biological function of carbon utilization, each pathway is activated by the presence of different sugars in the system. For example, the presence of arabinose activates the arabinose catabolism pathway and the presence of other sugars like ribose or maltose activates their respective pathways. Therefore, the entire carbon utilization system does not have to be coexpressed at the same time.

Investigation of this network shows a measurable coexpression within fibers and low coexpression across fibers, as evidenced by the correlation matrix C_{ij} in Fig. 20.5.

The activities of the genes in operons are reported individually in Ecomics, so we plot the activity of the individual genes. Genes in one operon are marked in the plot; for instance, the operon *fucAO* is unpacked as the two genes *fucA* and *fucO*. Synchrony within operons is given by the common polymerase reading, so it is trivial, hence not a proof of nontrivial fiber synchrony, although unpacking the operon creates a fiber. The test of fiber synchronization is then to compare the activity of any gene in the operon with that of the genes outside the operon. For instance *fucAO* with *fucR* and *zraR*.

Figure 20.4 (bottom) shows the associated functional network, obtained by thresholding

the correlation matrix of Fig. 20.5. A threshold $C_{ij} > 0.6$ was used to produce a functional network where a link between genes i and j exists if the Pearson correlation coefficient C_{ij} from Fig. 20.5 is above the threshold 0.6. Links in this functional network exemplify synchronization between genes in the fiber, according to their high pair correlation. The resulting functional network corresponding to the carbon circuit is shown in Fig. 20.4 (bottom).

There is a lot to unpack here. In the ideal case of perfect synchrony in the fibers, we would expect to see a functional network made of a clique for the genes in each fiber, and no connections between genes in different fibers or between fibers and regulators. This pattern would indicate perfect synchrony. We see this approximately, but not perfectly, in Fig. 20.4 (bottom). For instance, the fiber A working on ribose catabolism is almost a clique: fully connected internally, but it has a connection to a fiber D responsible for maltose catabolism. There is still some synchronization between fibers, but it is very weak.

The fibers B and C involved in arabinose and N-acetylglucosamine utilization are indeed perfectly synchronized. They form a clique in the functional network with no external links. We also observe that the regulators *crp*, *araC*, *exuR* and *gntR* are fully disconnected in the functional network, yet connect to the set of fibers in the structural network, as predicted.

Modularity algorithm provides the colors in Fig. 20.4 (bottom), which correspond exactly to the balanced colors obtained by fiber analysis of the TRN in Fig. 20.4 (top). This demonstrates the structure \rightsquigarrow function relation in the *E. coli* TRN.

The functional network shows a high correlation (synchrony) for genes in the same fiber, and a lack of correlation between genes in different fibers. The synchrony and lack of it observed in and outside fibers does not depend on whether the genes in the fiber/outside fiber are connected in the structural network.

A link in the functional network does not imply one in the structural network, that is, in the TRN, and vice versa. The most clear example of how different the functional network is from the structural TRN is the role of the master regulator *crp*. This regulator is clearly the hub of the entire carbon utilization circuit in the TRN of Fig. 20.4 (top), since it regulates all the genes in this circuit, together with many others that are not considered part of the carbon utilization. Yet in the functional network of Fig. 20.4 (bottom), *crp* is an isolated gene, not functionally connected with any of its regulated genes. That is, its regulon does not predict its functionality via synchronization. Indeed, while *crp* regulates a myriad of fibers, it does not synchronize with any of them.

Guilt by association techniques to annotate genes should be applied only to the functional network. Applying this approach to the structural TRN would give the wrong result, as shown for *crp*. In the multi-layer composite, fiber A in Fig. 20.4 (top), the genes *hemH*-*oxyS* are not linked with *add*; in fact, they are quite far apart in the network, separated by a distance of two steps, but they are still highly correlated and functionally related by synchronization, which is indicated by their connections in the functional network.

We conclude that the functional network does not directly reflect the structural network. These two networks measure different aspects of gene activity. The structural network is the network of physical interactions between genes via a TF, expressed by the source gene, binding to the promoter region of the source gene. A direct link here means direct physical

interaction. On the other hand, the functional network measures the functional association of two genes emanating from their synchronization and high coexpression.

These differences seem not to be clear in the literature, especially when the structural links are inferred directly from the functional ones, as is routinely done (Brugere et al., 2018; Horvath, 2011; Zhang and Horvath, 2005; Langfelder and Horvath, 2008; Bansal et al., 2007; Tegner et al., 2003; Liao et al., 2003; Wang and Huang, 2014; Marbach et al., 2012; Butte and Kohane, 1999; Maertens et al., 2018; Chen et al., 2008).

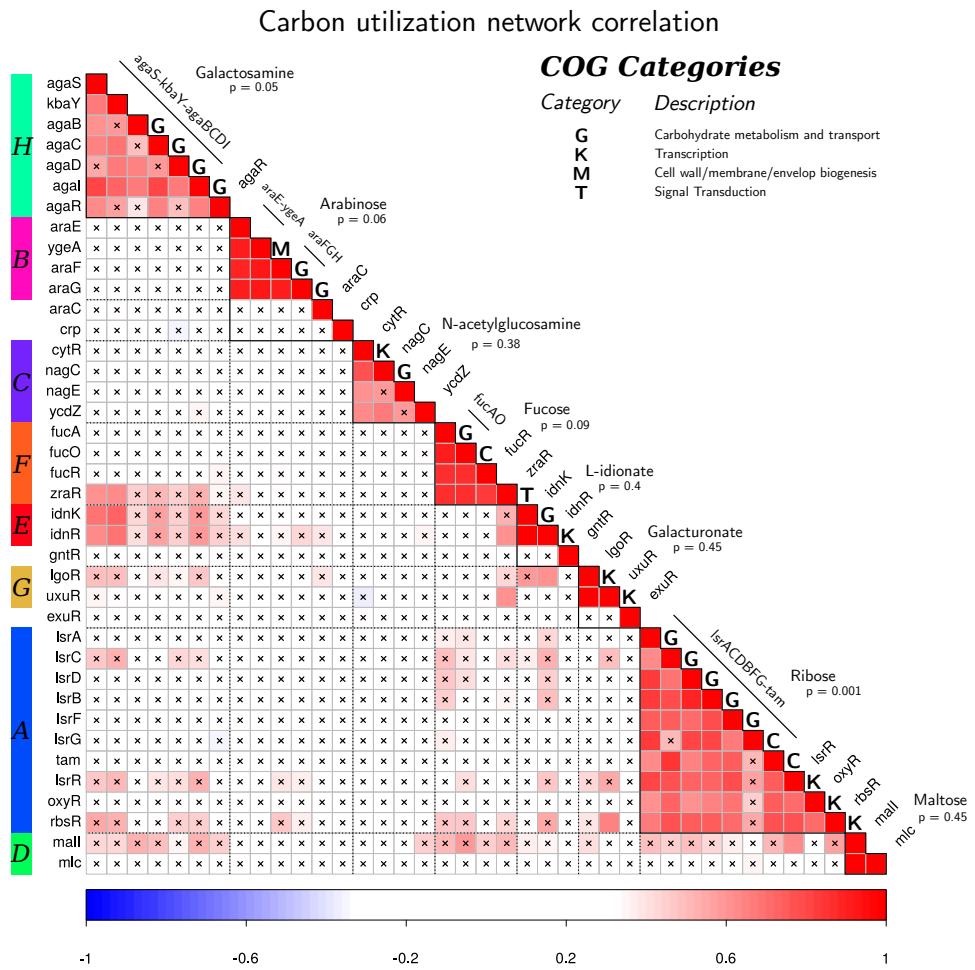
The only way to infer structural links from functional links or viceversa is using theory. In Chapter 22 we develop a method to infer the structural network from the functional one using fibrations, which resolves this issue. We will develop a symmetry-driven reconstruction of the structural network that guarantees that the resulting network reproduces the synchronization observed dynamically in the functional network. The method reconstructs links in the structural network independently of whether they are connected or not in the functional network.

20.4.3 Synchronization in the hierarchy of fibers in *E.coli* and *B.Subtilis*

Leifer et al. (2021) also present a study of coexpression patterns in *B. subtilis*, Fig. 20.6. The circuits presented include simple fibers: $|n = 0, l = 1\rangle$, $|n = 0, l = 2\rangle$, $|n = 1, l = 0\rangle$, $|n = 1, l = 1\rangle$, and $|n = 1, l = 2\rangle$; and multi-layer composite fibers in both species.

The observed correlations largely confirm synchrony within fibers, and lack of correlation between fibers. However, there are some interesting exceptions. For instance, the genes *cssR* and *cssS* in $|n = 0, l = 2\rangle$ in *B. subtilis* present large anticorrelations with a fiber (the multi-layer containing the gene *cgeC*). This anticorrelation may indicate extra transcriptional regulations between these fibers. These types of correlations can be used to guide the search for missing regulation edges, which are ubiquitous in genetic network reconstructions.

The existence of unexpected extra synchronization is also an indication of symmetries that are undetected in the TRN due to incomplete data. It is also observed that the multi-layer composite circuit example in Fig. 20.6 is overly synchronized. That is, rather than exhibiting no synchronization between the two layers of the building block (in the figure, the two fibers in the multilayer are indicated by names in red and green), the circuit exhibits complete synchronization among all genes. We hypothesize that this can be explained by the self-activation link at the gene *spoIID*, which would have turned this building block into $|n = 1, l = 0\rangle$, which is supposed to exhibit complete synchronization. Chapter 22 discusses an algorithm to perform this synchrony-based reconstruction of the TRN, guided by symmetries.

**Fig. 20.5**

Correlation matrix of carbon utilization network of *E. coli*. This is obtained by a filtering method based on the ICV. Operons are shown with a black line underneath the gene name along the diagonal. Correlations below 0.6 are marked with black crosses. UniProt database (Consortium, 2022) provides COG categories. The type of a fiber building block regulator defines the function of each block and is obtained from RegulonDB (Gama-Castro et al., 2016). Figure reproduced from (Leifer et al., 2021). Copyright © 2021, The Author(s).

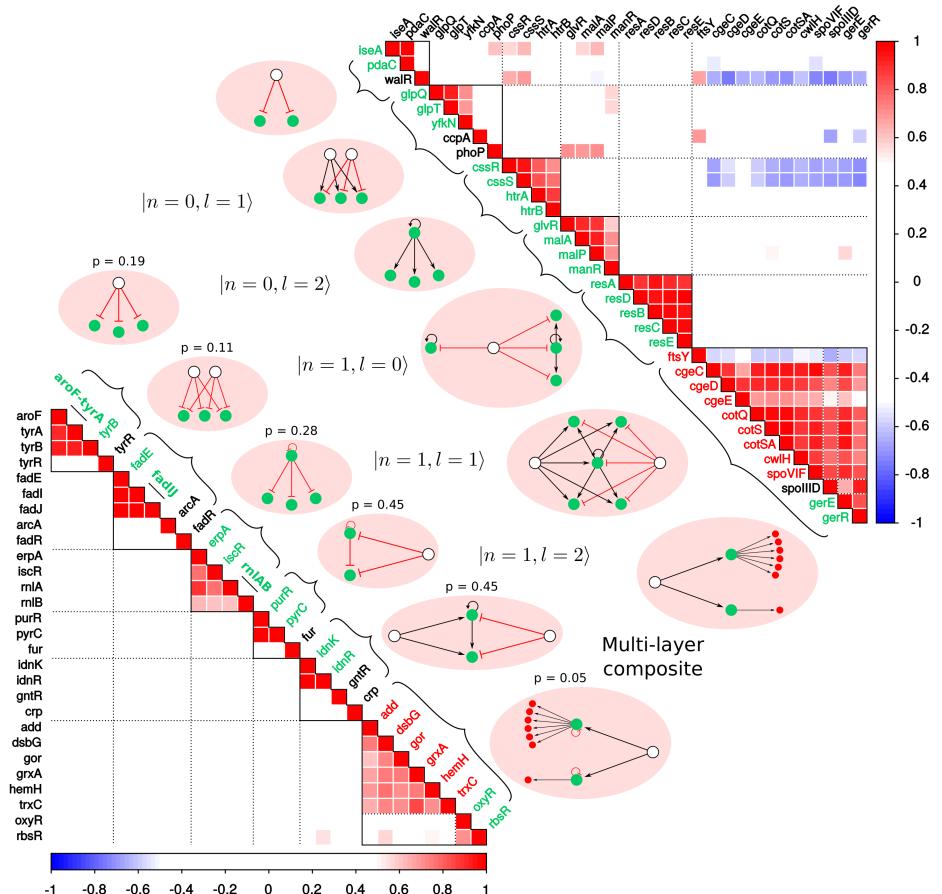


Fig. 20.6

Correlations in the hierarchy of fiber building blocks in *E. coli* and *B. subtilis*.

Fiber building blocks are represented as pink ovals. White nodes represent regulators and green and red nodes represent fibers. Black links represent activation and red links represent repression. Cross-correlations are shown with the lower diagonal (*E. coli*) and upper diagonal (*B. subtilis*) correlation matrices. Black lines indicate building blocks and black dotted lines indicate cross-correlations between different fibers. Operons are shown by a line underneath the gene names along the diagonal. Figure reproduced from (Leifer et al., 2021). Copyright © 2021, The Author(s).

All Biology Conspires Against Synchrony, so How Can We Be Alive?

Genes fibers of an idealized model exhibit perfect symmetry and exact synchrony. This is never realized in biology, as nothing is ideal in living systems. In fact, we expect that perfect synchrony and symmetry can easily be broken by the multitude of parameters and interactions, all different, of real systems. This leads to the question: how can a predictable pattern of synchrony, which is essential for survival, be achieved in biology? How can we survive in the presence of noise and disorder competing against symmetry? This question was vividly posed by Schrödinger (1944) in his seminal book, *What is Life?* Decades later, we still lack a definitive answer. While we do not claim to provide one here, we hope that the following speculations may contribute to the understanding of the issue.

21.1 Modeling in Biology

First, some remarks on modeling assumptions are in order. In this chapter—in fact, for much of the book—we start by working with idealized models that ignore many aspects of real biology. For example, when we say that two nodes are ‘synchronized’, we mean that their dynamic states are *identical*, not merely very similar. This, of course, is never realized exactly in practice, but it leads to ‘clean’ mathematics. It is also a standard modeling assumption in much of the synchronization community. Likewise, we work with ‘idealized’ ODEs based on uniform parameters and many simplifying assumptions that allow for their mathematical treatment.

In return for these simplifications, it becomes easier to understand the underlying mathematical principles. In later sections, we discuss more realistic modeling assumptions. It is worth bearing in mind that all models involve a trade-off. Simple models are easier to understand, but oversimplified models can be misleading. On the other hand, complex models may be more realistic, but they can be much more difficult to analyze and to understand. Moreover, complex models are often very sensitive to the exact assumptions being made: realistic ingredients are no guarantor of accuracy.

Ultimately, what makes a model worthwhile is not so much how realistic its ingredients are but how much insight it gives and how well its *predictions* corresponds to reality—what matters is not so much what is put in, but what we get out. For instance, astronomers regularly model the solar system as a collection of point masses. No actual planet is a point, and point masses cannot exist in reality; yet these models are accurate enough to get a space

probe close to its destination with high precision. Only when it has to land is it necessary to model the complex gravitational field and planetary terrain in greater detail.

21.2 Synchronization in biology is not ideal

One of the main advantages of using symmetries to characterize dynamics is that the synchrony patterns that we predict are combinatorial/topological properties of the network, thus largely independent of the explicit form of the dynamical equations. Therefore, we do not need to specify the equations that describe the dynamics of the system—be it a neural system, a genetic network, or any other physical, chemical, or biological system—beyond the general form from Definition 3.10, as long as the ODEs are admissible for the graph. We, therefore, derive all our results for a general type of dynamics, which can then be specialized to particular systems. However, particular systems are characterized by a large number of parameters and most of them are required to be the same for symmetry to exist. Unfortunately, this ‘uniformity condition’ is never strictly realized in biology. How is the coherent synchronization necessary for life to emerge?

We treat this question below. We first elaborate on an ideal model of gene expression that predicts perfect synchrony. Then we show that the conditions for this model are not realized in biology due to the broad parameter space of interactions among genes, proteins, and small molecules. We discuss how to reconcile this result with the observed (quasi-perfect) synchrony in real systems.

21.3 An ideal ODE model of gene expression dynamics

Gene expression dynamics in cells depend on a variety of processes including transcription, translation, protein folding, mRNA and protein degradation, and mRNA and protein dilution in growing cells (Karlebach and Shamir, 2008). All of these processes are stochastic and gene-specific, which prevents complete and precise synchronization of expression profiles. However, strictly speaking, synchrony via symmetries requires an idealized deterministic model of gene expression with not too many parameters, where, in general, all interactions and parameters are the same.

We first define this ideal model, where fibers and synchrony are perfect. Each gene has a variable describing the cellular concentration of the gene product, i.e. the protein encoded by the gene. Assuming short mRNA half-lives (and linear, unregulated mRNA degradation), transcription and translation can be effectively described by a single production rate, with the protein product concentration as the output variable. However, for reasons of data availability, we instead consider mRNA concentrations to be the relevant gene product. If the finite duration of transcription and translation is fast as compared to cell

growth and protein lifetime (or when describing cell steady states) it can be ignored. This justifies an ODE without time delay. The expression dynamics of the given gene depends on two contributions: degradation and synthesis (the arguments are independent of this assumption):

$$\frac{dx(t)}{dt} = -\alpha_x x(t) + f_x(y, z, \dots). \quad (21.1)$$

The input function f describes the transcription rate as a function of the active TF concentrations of the incoming genes. It can be modeled, for simplicity, by a Hill function of other's TF activities y, z, \dots . In this model, the internal dynamics is given by the first term and the interaction by the second.

By the Rigid Equilibrium Theorem (Golubitsky and Stewart, 2023, Theorem 14.9), a crucial ingredient for synchronization of the genes in a fiber is that the input functions are the same within the fiber. This is realized if the Hill functions mediated by a given TF are the same, i.e., determined by the same kinetic constants, which must be independent of the target gene. We elaborate on this modeling assumption and many others in detail below, using the FFF as a prototypic example.

21.3.1 A more realistic ODE model of the FFF

To understand the role of the parameters, we describe a quite general model where all the constants and parameters defining the dynamics are different. We use the FFF. As in (Karlebach and Shamir, 2008), the amount of product (measured by mRNA concentration) of the three genes in the FFF—namely X = external regulator to the fiber, Y = autoregulated TF in the fiber, Z = enzyme in the fiber regulated by Y —are described by dynamical variables $(x(t), y(t), z(t))$ obeying the system of ODEs:

$$\begin{aligned} \frac{dx(t)}{dt} &= -\alpha_x x + f_{\text{ext}} \\ \frac{dy(t)}{dt} &= -\alpha_y y + \gamma_y f_y(x, y) \\ \frac{dz(t)}{dt} &= -\alpha_z z + \gamma_z f_z(x, y). \end{aligned} \quad (21.2)$$

Here $x(t), y(t)$ and $z(t)$ are the time-dependent concentrations of gene products $i = X, Y, Z$ measured by the mRNA concentration in the cell, α_i is the mRNA degradation rate constant for gene i , and γ_i is the maximal synthesis rate of the mRNA product of gene i .

The input functions f_y and f_z of the genes Y and Z can be modeled by different forms, e.g. motivated by statistical mechanics models of TF binding based on binding states and their energies. In our simplified model we follow Kaplan et al. (2008), who have experimentally determined gene input functions for the carbon utilization system of *E. coli*, showing that the contributions are approximately multiplicative (AND gates) in many cases. Other gates can also be assumed: using OR gates instead of AND does not break the symmetry as long as the same (AND/OR) model is used for all genes in the fiber. Therefore we assume that the input functions of gene X and Z are a multiplicative function of the inputs from gene X

and gene Y:

$$\begin{aligned} f_y(x, y) &= f_{x \rightarrow y}(x) f_{y \rightarrow y}(y) \\ f_z(x, y) &= f_{x \rightarrow z}(x) f_{y \rightarrow z}(y) \end{aligned} \quad (21.3)$$

where $f_{i \rightarrow j}$ is the input function of gene j characterizing the binding probability of the TF expressed by gene i on the DNA binding site of gene j . Thus, for instance, the input function of gene Y is a logical AND function of the input function from gene X, $f_{x \rightarrow y}(x)$, and from gene Y, $f_{y \rightarrow y}(y)$ through the AR loop.

The existence of synchrony in this model of FFF requires a uniformity assumption: the functional forms of the input functions in the fiber must be of the same type. For instance, we cannot mix a Hill function in one link with a Boolean function in another. Typically (Karlebach and Shamir, 2008; Alon, 2019; Klipp et al., 2016; Kaplan et al., 2008), the functional form of the single input functions is fitted by Hill functions. Then we model all input functions assuming Hill interactions of the following form (Alon, 2019):

$$\begin{aligned} f_{x \rightarrow y}(x) &= \frac{x^{n_{xy}}}{K_{xy}^{n_{xy}} + x^{n_{xy}}}, & f_{y \rightarrow y}(y) &= \frac{y^{n_{yy}}}{K_{yy}^{n_{yy}} + y^{n_{yy}}}, \\ f_{x \rightarrow z}(x) &= \frac{x^{n_{xz}}}{K_{xz}^{n_{xz}} + x^{n_{xz}}}, & f_{y \rightarrow z}(y) &= \frac{y^{n_{yz}}}{K_{yz}^{n_{yz}} + y^{n_{yz}}}. \end{aligned} \quad (21.4)$$

These functions are determined by a set of kinetic parameters. In general, they can be written as:

$$f_{i \rightarrow j}(x_i) = \frac{x_i^{n_{ij}}}{K_{ij}^{n_{ij}} + x_i^{n_{ij}}}, \quad (21.5)$$

where x_i represents gene i , K_{ij} is the dissociation constant between TF i and the promoter DNA region of gene j (that is, the TF concentration at which half of the binding sites are occupied), and n_{ij} is the cooperative Hill coefficient of the input function of genes j representing the cooperative binding of the TF i at the DNA binding site of gene j . Finally, in (21.2), f_{ext} represents an external input to gene X that is irrelevant for the existence of synchronization in the FFF, and exemplifies the input regulation of gene X by the rest of the network. Genes Y and Z may also have outgoing links, but they do not appear in the dynamical equations because their effect shows up only in the dynamics of other nodes.

21.4 The non-idealized ODE model does not lead to synchronization

The dynamical system represented by (21.2)–(21.4) with all the kinetic constants different has no symmetries and no synchrony subspaces. That is, the graphical representation of the dynamical system (21.2) does not have any fibration symmetry; the input trees of distinct genes are not isomorphic. This is because an admissible graph representing these equations will have all the edges/arrows in the graph different since they represent different interaction terms in the ODE.

In the graph fibration formalism, two arrows are ‘the same’ if and only if the propagated

state of the source node is processed in the same way by the respective target nodes. Different types of arrows can occur, so the ‘output’ effectively depends on the target. Indeed, in this formalism, the only output from a node is propagation of its state to the target. However, different targets *process* this state in different ways, specified by the arrow types that determine the components of the ODE. Possibly different signals from a node c , depending on its state x_c , are packaged into different component functions f_c of the admissible ODE. Thus, arrows indicate the ‘influence network’: which nodes affect which, and how they do so. See Section 3.3.

Therefore, in principle, this dynamical system cannot sustain any synchronized activity of its genes (except in extremely special ‘non-generic’ circumstances). In the underlying FFF graph, all links have different strengths γ_i , and the same goes for the other kinetic constants that define the input functions. Different nodes would receive different input signals, destroying synchrony. Thus, it is clear that there can be no isomorphic input trees between the genes in such a graph. The Rigid Equilibrium Theorem (Golubitsky and Stewart, 2023, Theorem 14.9) makes this statement precise, with a rigorous proof.

21.5 The uniformity assumption: idealized FFF model leads to synchronization

Recall that the condition for the existence of isomorphisms between input trees is a local isomorphism in which nodes are mapped to nodes and edges are mapped to edges, one-to-one. In contrast, in the ODE (21.2)–(21.4), all parameters and input functions are different. In a network representation of the FFF circuit representing (21.2)–(21.4), each edge represents a different class of interaction given by their different weights γ_{ij} or different input functions $f_{i \rightarrow j}$. Therefore, in (21.2)–(21.4) an isomorphism cannot exist between Y and Z since the links cannot be mapped between the input trees representing the activity of the genes.

To obtain an isomorphism between input trees, corresponding links mapped by the isomorphism must be of the same type. In the case of TRNs, these links must represent the same strength of interaction and the same input function. We thus arrive at the ‘*uniformity assumption*’ required for the existence of perfect symmetry fibrations and perfect synchronization. To satisfy this condition we require an idealized system of equations where the single gene input functions and parameters (binding constants, synthesis strengths and degradation strengths) are the same for all genes that belong to a fiber. We do not require further equalities between parameters for distinct fibers. Specifically, we require:

- The strengths of the maximal synthesis rates of genes in the fiber are the same. For the FFF,

$$\gamma_y = \gamma_z \equiv \gamma. \quad (21.6)$$

The degradation constants are the same as well to guarantee the same internal dynamics, otherwise, nodes will have different types, breaking the symmetry:

$$\alpha_y = \alpha_z \equiv \alpha, \quad (21.7)$$

but they can be absorbed into the definition of γ_i .

- The Hill functions $f_{i \rightarrow j}$ producing the outputs of gene i are of the same type and are characterized by the same parameters. Thus, the single Hill function parameters in $f_{i \rightarrow j}$ depend only on the properties of the source i th TF, but not on the properties of the target binding site at gene j . In the FFF, for example, for the input trees of genes Y and Z to be isomorphic, we require the input functions $f_{y \rightarrow y}$ and $f_{y \rightarrow z}$ to have the same shape; a condition that is achieved by

$$K_{yy} = K_{yz} \equiv K_y, \quad (21.8)$$

and Hill cooperative coefficients

$$n_{yy} = n_{yz} \equiv n_y. \quad (21.9)$$

Additionally, the outputs of gene X must be the same:

$$K_{xy} = K_{xz} \equiv K_x \quad \text{and} \quad n_{xy} = n_{xz} \equiv n_x. \quad (21.10)$$

The outputs of X and Y need not to be the same, though, so in general $K_x \neq K_y$ and $n_x \neq n_y$.

Under these uniformity conditions, the AR link $Y \rightarrow Y$ has the same strength and type (input function) as the link $Y \rightarrow Z$, and X also regulates Y and Z in the same way. Only under these conditions can an isomorphism between Y and Z occur, and perfect synchronization between Y and Z arise in the FFF:

$$y(t) = z(t). \quad (21.11)$$

In general, for an isomorphism in a fiber to exist and to guarantee synchronization, it is required that for each link received by one gene in a fiber, there exists a link received by other genes in a fiber of the same strength and interaction type.

While this condition may seem very restrictive, we argue below that it is not difficult to be satisfied in TRNs if the interaction strengths between a TF and a DNA binding domain depend mainly on the source TF and less on the DNA structure of the promoter region. If so, we can drop the target-dependence on the coefficients in the gene input functions and use: K_y and n_y , and K_x and n_x .

Assuming this uniformity within fibers, we now consider an idealized ODE model with symmetrized gene input functions. The ODE (21.2) is then replaced by:

$$\begin{aligned} \frac{dx}{dt} &= -\alpha_x x + f_{\text{ext}}, \\ \frac{dy}{dt} &= -\alpha y + \gamma f_x(x) f_y(y), \\ \frac{dz}{dt} &= -\alpha z + \gamma f_x(x) f_y(y), \end{aligned} \quad (21.12)$$

with idealized input functions:

$$f_x(x) = \frac{x^{n_x}}{K_x^{n_x} + x^{n_x}}, \quad f_y(y) = \frac{y^{n_y}}{K_y^{n_y} + y^{n_y}}. \quad (21.13)$$

As discussed in Chapter 5, under these assumptions, genes Y and Z belong to the same fiber, and there is a synchronous solution $z(t) = y(t)$. It can be shown that this solution is

more stable and has a larger basin of attraction for the system than other solutions (DeVille and Lerman, 2015b; Nijholt et al., 2016), so it is highly probable that

$$\lim_{t \rightarrow \infty} (z(t) - y(t)) = 0. \quad (21.14)$$

So, under these uniform assumptions, gene synchronization is achieved via a symmetry fibration. It is still reasonable to ask: where on Earth could we find a system that satisfies all these conditions on the parameters so precisely?

21.6 Breaking of uniformity

In reality, the input functions will not be identical, leading to symmetry breaking in the synchronization of the expression profiles. There are many possible reasons for this. Below, we discuss some of them, and in Section 21.7, we argue how this uniformity assumption might be realized. This is fundamentally a modeling assumption, and like all such, it should be judged by whether its predictions can be verified, not by how closely it mimics reality. A map that is the same size as the territory is usually useless.

First, we enumerate what could go wrong for perfect symmetries in biological networks.

1. The parameters of the gene input functions depend, among other things, on promoter gene sequences and ribosome binding sites and can be expected to be similar between genes within a transcription unit (e.g., in an operon with a single promoter). Differences can be expected between genes in different transcription units (but belonging to the same gene fiber). Also, different AND/OR gates (Alon, 2019; Kaplan et al., 2008) and combinatorial gates can affect the existence of symmetries.
2. Our simplified model ignores many biochemical details (e.g. gene-specific mRNA lifetimes) that may alter the expression profiles of individual genes, and hence reduce correlations between different profiles.
3. Since gene expression is inherently stochastic, even larger deviations from coexpression are expected to arise at the single-cell level. Again, this particularly concerns genes that reside in one fiber, but in different transcription units.
4. We have assumed that mRNA production rates depend on TF activities via simple Hill functions. For each gene, the effects of different TFs are assumed to be multiplicative. The product of Hill functions represents a simple TF binding kinetics, and can be seen as a simplified version of the gene input functions (Bintu et al., 2005). Multiplicative gene input functions were experimentally observed in *E. coli* (Kaplan et al., 2008). However, the combinatorial nature of these interactions (Buchler et al., 2003) implies that many other gates, for instance OR gates, are possible (Mayo et al., 2006). Again, the main assumption to keep the symmetry fibration and consequent synchronization is that these gates are the same for genes inside the fiber, which may not be true in general.
5. Degradation was assumed to be linear and non-regulated, with identical degradation constants for all genes in the fiber. For concentration variables, degradation can effectively include dilution effects in growing cells. In fast-growing bacteria, and assuming

long-lived gene products, the degradation constant is approximately given by the cell's specific growth rate. The assumption of linear, non-regulated degradation of mRNA and protein has often been made in models, including NCA and the ODE models used to study the dynamics of network motifs (Alon, 2019; Liao et al., 2003). Typically, bacterial protein lifetimes are much longer than the cell cycle period, while mRNA lifetimes are still considerably smaller. Long protein lifetimes have been shown experimentally in *E. coli*, (Klipp et al., 2016).

6. Importantly, the activity of TFs can be modulated by ligands whose concentration carries information about the state of the cell. In the TRNs studied here, these additional regulations are ignored. Considering them may lead to a further subdivision of gene fibers, and to a predicted desynchronization of genes that are currently predicted to be synchronized. These effectors are obtained by searching for bacterial growth experimental conditions where fibers are activated, as explained below.
7. Our network reconstructions are certainly incomplete, leading to errors in fibrations and to mispredictions about coexpression. Adding another TF to the network may destroy the symmetry of a fiber and lead to its subdivision, entailing a change in predicted coexpression patterns.
8. We consider the TRN in isolation. In reality, it is part of a larger network consisting of several layers (including transcription, translation, protein interactions, and metabolism, with different types of regulation arrows connecting them). Fibration analysis of such networks, acknowledging the tight feedback regulation between transcriptional regulation and other cellular systems, might bring additional insights about the coexpression of genes in the context of an entire cell. An extended fibration theory of integrated biological networks is still in its infancy.

For all these reasons, synchronization of gene expression can only be partial, if it exists at all. It can be asked: Why do we start by assuming idealized symmetries across all the Part I of this book, just to later conclude that these symmetries may not exist in reality? Our answer is that this procedure—deriving basic, general laws under simplicity assumptions and treating deviations from them as necessary but small corrections—has been a very successful strategy in physics and has some mathematical justification. A biological example occurs in the repressilator (Elowitz and Leibler, 2000), a synthetic genetic circuit. Here, the idealized model has perfect \mathbb{Z}_3 symmetry, leading to a rotating wave state in which all nodes have the same periodic waveform, with each node being one-third of a period out of phase with the previous one. If the symmetry is broken by the different parameters of the ODE—even strongly—the waveforms become different. Still, their peaks remain very close to one-third of a period out of phase with each other. On the whole, phase relations seem less sensitive to perturbations than amplitudes are.

Since detailed models of gene expression (including precise gene input functions and covering all possible effectors) are lacking, we take idealized symmetric input functions as a working hypothesis to be tested experimentally *a posteriori*. If this hypothesis is accepted, we can model TRNs where different genes in a fiber are modeled with the same gene input function. This theory predicts the exact coexpression of genes in each fiber. Deviations

from this behavior can then be interpreted as ‘weak symmetry breaking’, which could be treated by perturbation theory, allowing for additional variation in gene expression.

In this picture, we first imagine a hypothetical—biochemically possible—cell in which all assumptions are justified and in which perfect coexpression is realized. We then consider the real cell to be a ‘weakly symmetry-broken’ version, possibly evolved to achieve specific patterns of non-coexpression for genes that otherwise would be symmetric. Such symmetry breaking occurs, for example, if two transcription factors in a fiber are differently regulated by different interaction strengths. The resulting subdivided fibers allow for more diverse and adapted expression profiles while retaining coexpression under certain conditions.

21.6.1 How physicists deal with this problem

Variations from ideal symmetry are ubiquitous in nature. As discussed in Section 18.4, there are two major types of symmetry breaking in physics: spontaneous and forced (or induced). Spontaneous symmetry breaking occurs when the model remains symmetric, but a fully symmetric state becomes unstable so that a less symmetric state bifurcates. There is an extensive mathematical theory of pattern formation via spontaneous symmetry breaking; see, for example (Golubitsky and Stewart, 2003). Forced symmetry breaking occurs when the model itself becomes asymmetric, for example, by parameters ceasing to be exactly equal. This type of symmetry breaking has also been widely studied.

The kind of symmetry breaking we are dealing with in this chapter is of this forced kind, and it is not the same as that discussed in the electronic circuit analogy in Chapter 18. That symmetry breaking is discrete and breaks completely the functionality of the circuit by adding different regulators to a symmetric circuit. The present breaking is dynamical and occurs smoothly as the different parameters of the ODE model are moved away from the uniform assumption of all being equal. This dynamical loss of synchronization by different parameters can be treated by perturbation theory, and it does not represent a new principle of functionality of the cell. It is just that cells are noisy and we must take this into account.

Variations from exact symmetry can be found in all theories that are computable in physics (Landau and Lifshitz, 1977), including the theory of the Higgs boson and the discovery of new hadrons by Murray Gell-Mann (Weinberg, 1995; Gell-Mann, 1995; Georgi, 2018). Assuming symmetries but acknowledging that they are broken has been a typical step in many theoretical physics discoveries. In a theoretical model, we first identify the exact symmetry and then proceed by perturbation expansion to describe the real system. This is a successful theoretical procedure in physics, and we believe that it could bring fruitful results to biology.

Indeed, we could argue that no symmetry is exactly realized in nature, or is not broken. Even in the SU(3) flavor symmetry of chromodynamics in the Standard Model of particle physics, the constituent masses of quarks up, down, and strange are not perfectly symmetric ($m_u = 336$ MeV and $m_d = 340$ MeV, $m_s = 486$ MeV). Despite this asymmetry between masses, by assuming perfect symmetry (all masses are the same), we can put the three quarks into a triplet (which assumes that they can be interchanged by an SU(3) symmetry transformation) and predict the existence of all the composite hadrons in the octet of Gell-Mann—such as protons, neutrons, and the heavier baryons—with high precision (Gell-

Mann, 1995; Weinberg, 1995; Georgi, 2018). The remaining small physical differences in quantum behavior arising from the asymmetries are then treated with perturbation theory from the ideal symmetric case. In biology, this perturbative theoretical approach can be readily translated to describe the cell's biological network's departure from ideal symmetry, which can be quantified and made computable by perturbation theory, and therefore, point to the efficacy of the perturbation regime.

We elaborate below on how this symmetry breaking in gene fibers can be handled, at the level of perturbation theory, in the 'weak symmetry breaking regime'. If the heterogeneities between gene input functions are small, they can be handled mathematically by a perturbation expansion around the symmetric state.

Variability is an intrinsic property of biological systems and a main driver for phenotypic change among organisms and species. Biological variations by symmetry breaking may explain phenotypic variations since ideal symmetries must be broken for evolution to exist.

If a biological system is close enough to the idealized symmetry, we can invoke conditions of hyperbolicity (Guckenheimer and Holmes, 1983), discussed in Chapter 8, to infer that robust approximately synchronous patterns of expression exist. A theory of perturbation applied for approximate fibrations has been developed by Coram and Duvall (1977). This theory shows that a perturbative expansion can be rigorously controlled, and the 'loop' corrections (referring to theoretical methods used to deal with perturbation theory) can be analytically computed, in principle, to any order.

This is stated in Theorem 2.6, page 282 in (Coram and Duvall, 1977): 'If $p : E \rightarrow B$ is an approximate fibration, and if E and B are manifolds, then for every $b \in B$, F_b satisfies the small loops condition.' Thus, approximate fibrations are computable and are backed up by rigorous mathematical results in the framework of category theory. Experimental evidence, e.g. Fig. 20.5, suggests that symmetry breaking must be weak because weak symmetry breaking from the input functions implies weak breaking of synchrony. An extended theory of quasi-fibrations developed by Boldi et al. (2022), and pseudo-balanced colorings developed by Leifer et al. (2022) and discussed in Section 13.5, can characterize biological networks beyond the idealized cases presented here. We have proposed an analogous theory of pseudosymmetry groups to explain pseudosymmetries in the *C.elegans* connectome in (Morone and Makse, 2019).

It should be recalled, though, that fibrations form groupoids, not groups. Such a theory would complement and augment the validity of the present approach to networks that may not be as completely characterized as the bacterial networks studied here. A pseudofibration theory would be especially handy to describe symmetries in human biological networks that are notoriously less complete and, of course, more complex than bacterial networks. Mathematical efforts in this direction are welcome. The next chapter discusses some of them.

21.7 Is symmetry necessary in biological networks?

From a mathematical point of view, the answer is ‘yes’. If there is robust and stable synchronization in a system with a hyperbolic equilibrium, then the only way to obtain such synchronization is via the fibration symmetries in the underlying network. The Rigid Equilibrium Theorem (Golubitsky and Stewart, 2023, Theorem 14.9) proves that there is no other way to obtain robust synchronization other than with (fibration) symmetries; see Chapter 5.

In general, dynamical states of synchronization without any underlying network symmetry is ‘fragile’ in the sense that the states are destroyed by small perturbations of the ODE that respect network structure (Golubitsky and Stewart, 2006). Approximate synchrony—as is typically observed in reality—need not be destroyed. However, approximate synchrony is different from a complete lack of synchrony, and it requires explanation. The simplest explanation is the approximate symmetry of the network, which takes us back to idealized models.

More robust synchrony can be imposed by making additional modeling assumptions. A common one is to assume there is a distinguished rest state 0 and impose the condition that the function f in the ODE $\dot{x} = f(x)$ satisfies $f(0) = 0$ —which happens, for instance, in 8.28 because the coupling assumed is generalized diffusive. In such cases, all nodes can synchronize at 0. However, in the absence of such special constraints, robust synchrony can only be achieved through symmetries in the underlying network. Armed with this mathematical result, we now elaborate on the experimental evidence for symmetries and synchronization in biological networks.

We address how to overcome the limitations of the application of the fibration model to biological network, as described in Section 21.6. The following discussion is very limited and should be seen as an initial speculation, definitively not definite, about the problem.

Our ODE model of gene regulation is a simplified one, as discussed in Section 21.6, many biochemical details are missing and identical gene input functions are assumed for all the outgoing links from a given gene. Since none of these assumptions is valid in reality, can we still claim that symmetries in network structures entail symmetries in gene expression? Instead of making this claim, we assume that these idealized symmetries will always be broken in reality, which precludes exact synchronization and introduces additional variation in expression. The exact symmetries of an idealized network is interpreted as predicting *approximate* symmetry/synchrony in the real biological system modeled by that network.

What are the main factors that lead to such symmetry breaking? Do observed gene expression patterns, despite these effects, still display a vestige of the unbroken symmetry? And, since synchronized expression patterns are beneficial and widespread in biological systems, could there be evolutionary pressures that counter symmetry breaking?

21.7.1 Weak symmetry breaking by differing gene input functions

To be coexpressed precisely, genes in a fiber must share the same regulatory input functions as in the idealized models (21.12) and (21.13). In contrast, differences in these functions or their quantitative parameters, as in (21.2) and (21.4), lead to weak symmetry breaking, i. e., incomplete coexpression. The simple input functions used in our model describe TF binding, but if we were to fit them to expression data, they would actually capture the entire process of transcriptional and translational regulation, including details such as mRNA stability, ribosome binding, and protein stability. Given all these details, our effective ‘gene input functions’ would certainly differ between genes (with the exception of genes in the same operon in bacteria) even when the remaining differences occurring post-transcriptionally may be negligible. In all other cases, there is no biochemical reason why gene input functions should be identical.

Thus, on the one hand, in reality, we can expect symmetry breaking to cause genes in a fiber to become desynchronized. On the other hand, we observe measurably higher coexpression within fibers as in Figs. 20.5 and 20.6. Hence, if gene input functions within fibers are found to be significantly similar; this may indicate selection pressures opposing potential symmetry breaking, and would, at the same time justify gene fibers as a theoretical tool of analysis.

21.7.2 Similarities between measured gene input functions

We first recall that perfect synchronization in a fiber requires only conditions on the input links to the genes in the fiber. This means that a given input function’s shape and parameters should depend on the source gene.

This point is discussed in Section 21.5 from the conditions leading to symmetrized equations (21.12)–(21.13). Biological evidence for this kind of shared input function among genes can be found in the work of Kaplan et al. (2008). They measure the input functions of genes in the well-characterized gene regulatory system of sugar catabolism in *E. coli*. Direct measurements of gene input functions, describing the effects of effector molecules (acting on a transcription factor) on subsequent gene expression, have been performed.

The observed input functions were quite diverse, but a careful analysis of the data in (Kaplan et al., 2008) reveals that the reported diversity does not affect the existence of fibration symmetries. On the contrary, the data support the existence of symmetrical input functions in precisely those genes needed for the fibrations to occur. As we saw in Section 21.5, the requirement of uniform input functions for fiber synchronization is that the input function of a given TF, or a pair of TFs that regulate a particular set of genes, depends on parameters of the input functions of the source TFs and not on parameters specified by the binding sites of the target genes. That is, the input functions over a set of genes that are regulated by the same TF are expected to be approximately similar, even though other input functions of other TFs might be different. This may not be satisfied in real systems and may lead to weak symmetry breaking from the predicted fibers.

This situation is studied in (Kaplan et al., 2008, Fig. 2), discussed in (Klipp et al., 2016,

Fig. 9.18) and reproduced here in Fig. 21.1. In this figure, the same cAMP levels are used in all input functions, and the same sugar levels are used in each column. Promoter activity is the rate of GFP fluorescence accumulation per OD unit in exponential phase. The figure shows promoter activity normalized to its maximal value for each promoter. Rows arranged by biological role. This figure shows that the input functions of *E. coli* sugar genes, regulated by the same TFs, are similar, even though the claim in (Kaplan et al., 2008) is that they are diverse and quite different. This claim is correct when we compare all of the input functions across different TFs. But when we look at them per TF to see whether the input functions are similar for a given TF, i.e., column by column in Fig. 21.1 (bottom), then the similarities are evident.

We can observe this behavior in the arabinose circuit of Fig. 21.2. Arabinose is the molecular effector that activates AraC. AraC is also induced by *crp* (which is activated by a small molecule cAMP) and regulates the catabolic enzymes *araBAD* and the transporters *araE-ygeA*, *araFGH*, *araJ* that bring the sugar inside the cell. The first column of panels in Fig. 21.1 shows the set of input functions measured experimentally as a function of the activator effectors cAMP and arabinose. The input functions for the genes *araBAD*, *araE-ygeA*, *araFGH*, *araJ* and *araJ* are similar, as noted in (Kaplan et al., 2008). That is, the input functions of the target genes depend mainly on the source genes, *araC* and *crp*, but are the same across the target genes: *araBAD*, *araE-ygeA*, *araFGH*, *araJ* and *araJ*. The input functions can be factorized into simple functions of Crp and AraC (21.2).

Figure 21.2 reproduces some of the results of (Kaplan et al., 2008). The circuit of the fiber *araE-ygeA*, *araFGH* with external regulators, Crp and AraC is shown in Fig. 21.2a. The genes in the fiber and regulators are not contiguous in the genome: *araE-ygeA* is located at position 2,980,764 in the genome and *araFGH* at 1,985,179; their regulators are at 70,387 (*araC*) and 3,486,120 (Crp), so the similarities of input functions are not due to shared promoters or short-distance effects in the genome (Junier and Rivoire, 2016).

The parameters defining the Hill functions, obtained in (Kaplan et al., 2008), are shown in the table of Fig. 21.2c with the shapes of the Hill functions plotted in the figures. The Hill input functions of *araE-ygeA* and *araFGH* are almost identical, and the coefficients for both Hill functions are equal within error bars. For instance, the cooperative coefficients $n_1 = 3 \pm 1$ and $n_2 = 2.4 \pm 0.4$ are within error bars, and so are the other coefficients—except for the dissociation constants $K_1 = 0.8 \pm 0.6$ (mN) and $K_2 = 0.2 \pm 0.1$ (nM), which are close to but slightly beyond the error bars.

The Hill functions fitted with these parameters are shown in Fig. 21.2b, supporting the conjecture of symmetrically related Hill functions in the fiber. The input functions can be factorized into simple functions of Crp and AraC (Fig. 21.2b), as assumed in (21.12)–(21.13). The effects of CRP and AraC are multiplicative on each regulated gene, and each multiplicative contribution is well described by a Hill function of the type used in the ideal model.

This result is corroborated by other gene input functions. For example, the input functions in the galactose system show strong pairwise similarities (see the second column in Fig. 21.1). The input functions of the operon *galETKM* and the gene *galP* are similar to each other. Likewise, the input function of *galS* is similar to that of *mglBAC*, yet they are quite different from the *galP* functions. These similarities are conditions needed for synchronization

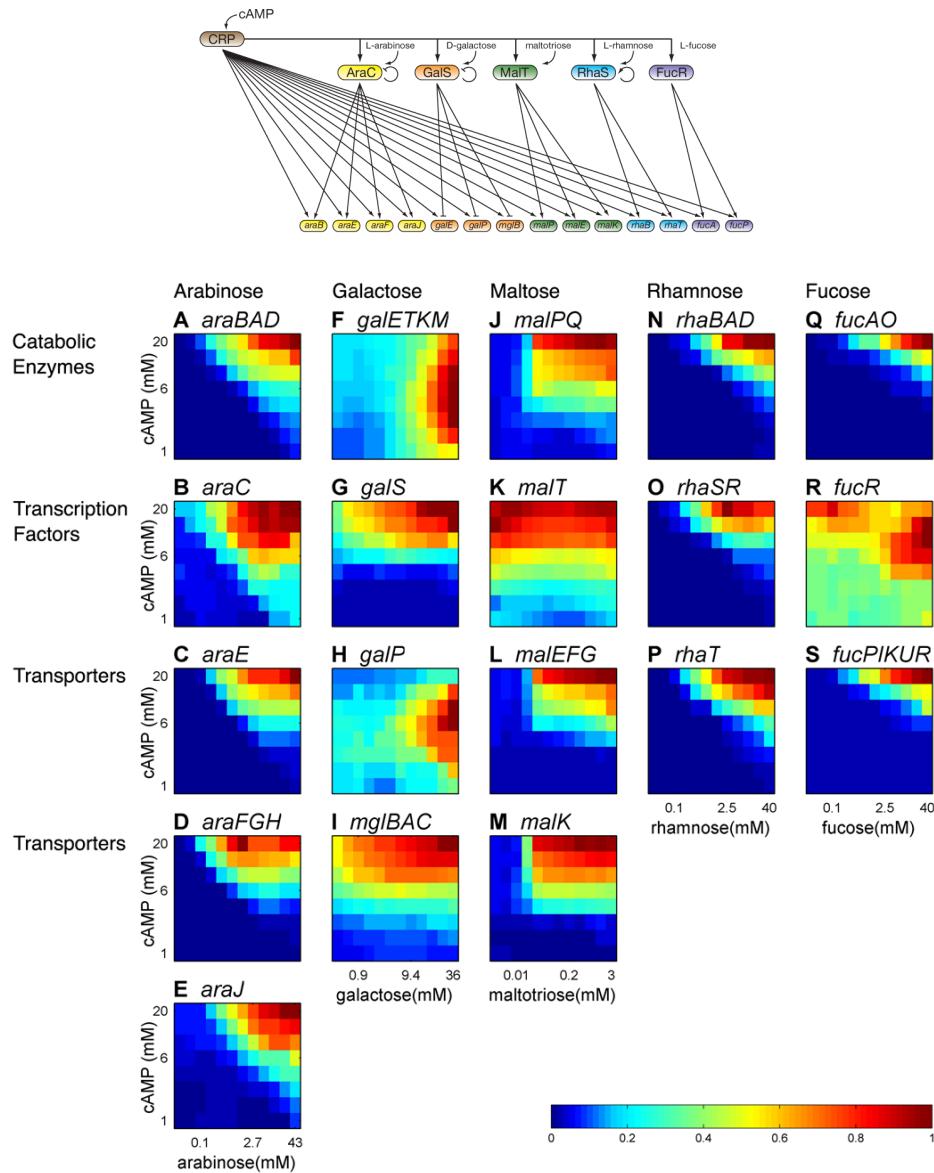
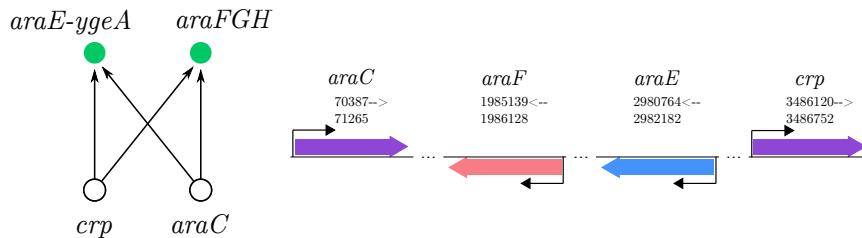
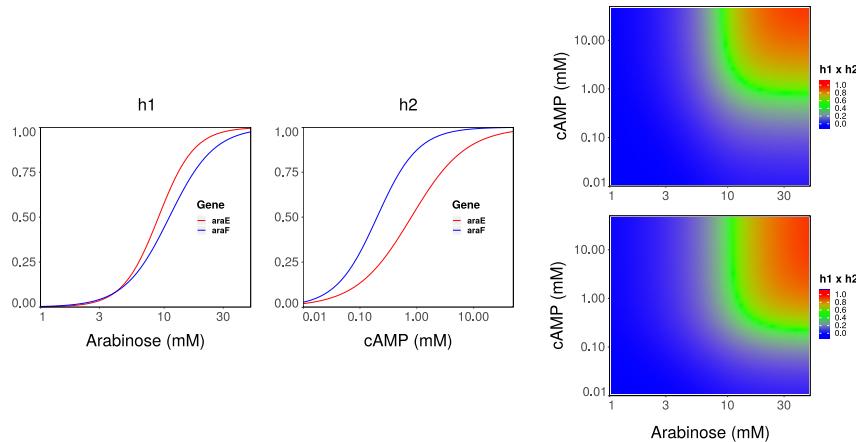


Fig. 21.1 **Study of input functions by (Kaplan et al., 2008).** Top: Regulon of *crp* in the *E. coli* sugar consumption network. Bottom: Input functions of *E. coli* sugar genes. (A–E) arabinose system; (F–I) galactose system; (J–M) maltose system; (N–P) rhamnose system; (Q–S) fucose system. Input functions are defined as the promoter activity at each of the 96 combinations of the two input signals, cAMP and the sugar. The x- and y-axes correspond respectively to sugar and cAMP concentrations in mM. Figure and caption reproduced from (Kaplan et al., 2008). Copyright © 2008, Elsevier Inc.

a Fiber



b Fitted input-function [Kaplan et al [38]]



c Parameters of the arabinose input functions measured in Kaplan et al. [38]

Operon	k_1	n_1	k_2	n_2	rms fit error	Description
<i>araE</i>	0.8	0.9	9	3	0.05	MFS transporter, low-affinity
<i>araFGH</i>	0.2	1.2	11	2.4	0.06	ABC transporter, high-affinity

Fig. 21.2

Input function study of the arabinose consumption circuit from (Kaplan et al., 2008). (a) Circuit producing arabinose with its fiber in green. (b) Input functions measured experimentally and modeling in (Kaplan et al., 2008). (c) Coefficients measured for the Hill input functions of *araE-ygeA* and *araFGH* promoters. Data reproduced from (Kaplan et al., 2008). Copyright © 2008, Elsevier Inc.

in the galactose fiber, which is split in two. Likewise, the remaining carbon circuits in Fig. 21.1 suggest the isomorphisms in the input functions needed for the symmetry fibration. In the maltose circuit, the input functions of *malPQ*, *malEFG*, *malK* are similar, yet different from *makT*; in the rhamnose circuit, *rhaBAD*, *rhaSR*, *rhaT* are all similar; and in the fucose

circuit, *fucAO* and *fucR* are relatively similar, explaining the synchronization observed in this carbon circuit in Fig. 20.5.

Thus, the observed Hill input functions in this circuit are consistent with uniformity conditions for a symmetry fibration. These experiments support the existence of local equivalences between genes in a fiber through uniform input functions. Eventually, the effectiveness of the fibration in capturing synchronization in gene coexpression should be validated experimentally by coexpression pattern analysis.

21.7.3 TF binding sites

What could be the molecular origin of the similarity in the Hill input functions observed above? Below, we offer a tentative answer to this question, which could lead to ideas for experimental and analytical tests of the molecular origins of the fibration symmetries. We elaborate on whether the mere existence of binding motifs in the DNA sequences of binding sites (these motifs are in the DNA sequence and should not be confused with network motifs) could lead to similar gene input functions, as required by symmetry. In principle, Mayo et al. (2006) have shown that a gene input function can be changed very flexibly by only small changes in the sequence. In principle, these results could already rule out a molecular origin for symmetric input functions such as binding motifs.

Indeed, the observed similarity is surprising because gene input functions can depend on minute changes in the binding site sequences (Mayo et al., 2006). What determines a gene input function is a number of factors that control the binding and unbinding properties of the regulators to DNA (Klipp et al., 2016). Likewise, promoter strengths (and ribosome binding site strength) differ over orders of magnitude (Zelcbuch et al., 2013). Furthermore, different TF binding sequences allow for a range of binding strengths, corresponding to a wide range of effective parameters in the gene input functions.

However, it seems likely that there should be a molecular explanation of the symmetries as well. Confirming or denying this view would require an analysis that compares TF binding site sequences or promoter sequences across genes, within and between fibers, and that shows significant differences between the two cases. Such an analysis is beyond the scope of the present book. So, as it stands, the arguments below are merely conjectural. The molecular basis of Hill functions has been described in (Mayo et al., 2006), and it would be an interesting problem to reconcile these results with the uniformity assumptions required for symmetries and synchronization that we impose here.

Among the different factors that determine a Hill function, below we analyze existing data on the binding DNA motifs of the TF at the promoter's nucleotide sequence (Klipp et al., 2016) that may lead to similar gene input functions. A given TF will bind preferentially to promoter sequences that contain well-defined TF-binding motifs. These motifs are used in bioinformatics analysis to determine the binding probability of a TF to DNA through position-specific scoring matrices (PSSM). Such matrices are constructed using statistics on binding sites' DNA sequences, characterized experimentally for a given TF (Hertz et al., 1990; Medina-Rivera et al., 2011). Evidence is seen in experimental studies of DNA motif analysis of TFs, showing enriched sequence-logo binding site DNA sequences. See, for

instance, the TrpR repressor for tryptophan biosynthesis in *E. coli* logo analysis (Medina-Rivera et al., 2011).

We analyze the sequences of the binding sites for the arabinose circuit controlled by the TF AraC, whose gene input functions are shown in the first column of Fig. 21.1. The TF AraC binds to two promoters (it binds to more but we study only two). One binding site is promoter araEp, which controls the expression of the gene *araE-ygeA*, which expresses the proteins AraE and YgeA. The other binding site that we study is promoter araFp which expresses the operon *araFGH* producing proteins AraF, AraG, and AraH. The sequence, positions, and features of these two promoters are shown in Fig. 21.3 (top). There are two promoter sequences for araEp and four for araFp. Figure 21.3 (bottom) shows the sequence logo analysis from (Crooks et al., 2004) of these sequences, revealing enrichment at positions 1, 2, 4, 6, 7, 8, 9, 13, 14, 15, 16, 17.

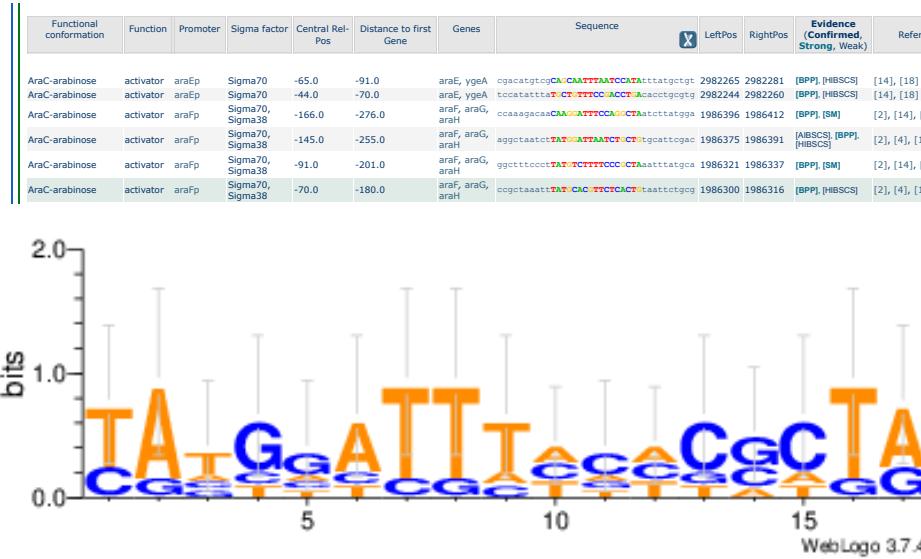
Is the similarity in these DNA sequence motifs evidence for the similarity of the input functions of AraC at these promoter sites? First, we see whether this enrichment similarity is consistent with the similarity in input functions across the first column in Fig. 21.1 obtained in (Kaplan et al., 2008). Here we just point out that the existence of these well-defined DNA binding motifs is consistent with (Kaplan et al., 2008) since the input functions are quite similar; for instance, compare panel C (*araE*) and panel D (*araFGH*) in Fig. 21.1.

Are all these results also consistent with the conditions needed for the validity of symmetry fibrations and the concomitant gene coexpression observed experimentally? This is not definitive. Many other factors affect these binding strengths. But this partial evidence points to an affirmative answer. More studies are needed to determine the molecular underpinning of these symmetries. We conjecture that, eventually, the synchronization and fibration symmetries we observe at the TRN level might be explained by symmetries underlying the DNA sequences that control the binding and unbinding of molecules to DNA for their regulation. This could shed light on the structure of the gene regulatory code as discussed next.

21.8 Selection pressures for symmetry

Despite all the possible molecular biological reasons for symmetry breaking, our study demonstrates gene coexpression within fibers as long as meaningful sets of biological samples are considered (i.e., samples in which the genes are significantly expressed). For gene pairs within a transcriptional unit, this coexpression would not be surprising. However, we found coexpression also in genes that are controlled by different transcription factors. Thus, despite inevitable asymmetries among gene input functions, the observed symmetry breaking is not very strong. There are two possible explanations. First, gene input functions (when normalized to the typical TF levels) may just not be very diverse, so network structure alone may determine coexpression patterns. However, there may also be a second, functional explanation, by which gene input function would be specifically similar within gene fibers.

While some gene input functions in *E. coli* mapped by (Kaplan et al., 2008) show diversity ranging between approximate AND and OR gates, depending on relatively small variations

**Fig. 21.3**

Logo analysis of the binding sites. We perform a logo analysis using (Crooks et al., 2004) at promoters araEp and araFp for the TF AraC. **(a)** Sequences of binding sites of the promoters of *araE-ygeA* and *araFGH*. **(b)** Sequence logo analysis of these binding sites *araE-ygeA* and *araFGH* shows strong similarities between the two promoters, as demonstrated by the enrichment at several DNA sites.

of the promoter sequence, the diversity can be grouped as shown in Section 21.7.2, and this grouping agrees with the conditions needed for the existence of a fibration.

What might the evolutionary role of this similarity be? Evolutionary pressure for robust gene coexpression would lead not only to the emergence of fibers (which support partial coexpression), but also to convergence of gene input functions within gene fibers during evolution, to make this coexpression more precise. The gene input functions of sugar genes reported by Kaplan et al. (2008) point in this direction: the genes activated by the regulator AraC (although belonging to different operons) show consistently similar responses to arabinose and cAMP, indicating similar gene input functions inside the fiber. The same holds for the gene sets activated by TFs in the other fibers in the circuits considered, GalS, MalT, RhaS, and FucR. Strikingly, the self-activator RhaS shows the same expression response as its target genes, exactly in line with the fact that they belong to one fiber.

Finally, since edges in a graph are merely proxies for gene input functions, and since the effects of TFs may gradually change during evolution, the appearance and disappearance of edges, and the evolution of gene input functions themselves are just two descriptions of the same process. We may even extend this conclusion. If selection pressures can make gene input functions within a gene fiber converge for synchronization and function, we may speculate that the same pressures that adjust gene input functions may also act towards adding more genes to a fiber by adding or removing input arrows. The evolution of network

structures and input functions (and even of gene function itself) may go hand in hand. For example, if gene products are involved in the same process, selection for coexpression may first put them into one fiber and then make their gene input functions converge (even before these genes may happen to be placed in one operon).

21.9 Towards a gene regulatory code

The *genetic code* is a set of rules engraved on the DNA sequence used by the cell to translate information from DNA into proteins. The ‘letters’ of this code are the codons that consist of triplets of base pairs.

While the genetic code determines the amino acid sequence of the protein from the DNA of genes, other parts of DNA, like the binding sites of the TFs, determine where and when these TFs bind to produce more proteins. The machinery that expresses these genes is controlled by the *gene regulatory code*. In turn, this regulatory code must be engraved in the structure of the gene regulatory network, which has fibration symmetries.

The full information flow goes from DNA to DNA. The genetic code transmits information from the DNA sequence of the gene expressing the protein to the amino acid sequence of the protein. This protein is a TF that binds to another DNA sequence at the promoter site of the target gene, expressing another protein. The information is then transferred from the TF to the new protein expressed at the target site.

The transfer of information is then between two DNA sequences, that of the source gene and the binding site of the regulated gene. This information flow is represented by an arrow in the regulatory graph, and all the arrows together build the fibration-rich regulatory code. It would not be surprising if we were to find symmetries in these sequences as well, which would reveal the regulatory code underlying gene regulation.

From Function to Structure: Network Inference Reconstruction

Chapter 20 deals with the structure \rightsquigarrow function relation by testing how the fibration symmetries of the structural transcriptional network determine the functional synchronization observed in gene coexpression. This validation largely relies on the completeness of the structural network. But real structural networks in biology are rarely complete. On the other hand, the functional network can be obtained for a fairly large number of biological nodes and, in many cases, for all of them, e.g., gene coexpression patterns of entire organisms. We use this information to infer the structural network of connections using a symmetry-driven reconstruction algorithm guided by balanced colorings obtained from cluster synchronization of the biological units. Thus, the present chapter deals with the inverse problem: how to use function \rightsquigarrow infer structure. The algorithm reconstructs incomplete gene regulatory networks from coexpression patterns as well as incomplete connectomes from neural synchronization data. This represents a crucial application of fibration theory to understanding biological pathways, with potential applications for reconstructing pathways for drug development and the understanding of disease.

22.1 Biological networks are never complete

The concept of symmetry is at the basis of physics, but we have seen that it is not (yet) as pervasively adopted in biology. Symmetries in biological networks have been very hard to find since the time of Monod's first formulation of the symmetry problem in regulatory networks (Monod, 1970). In our opinion, three crucial aspects have contributed to determining the reduced attention to the role played by symmetries in biological systems:

- Historically, the symmetries of a network were identified with automorphisms, a very rigid form of symmetry, rarely observed in biology. Furthermore, automorphisms are not the right tool to describe the notion of synchronization since they miss synchronies that only fibrations can identify.
- Even playing with the more general symmetry of fibration, biological systems are never exactly symmetrical: the high redundancy of biological systems leaves space for imperfections and small asymmetries. These do not affect the overall working of the system, but fail to be captured by a rigorous mathematical definition of exact fibration.
- Experimental observations providing the basis for constructing biological networks are inherently imperfect and incomplete, because biological systems are noisy and dis-

ordered. Therefore perfect symmetries are never realized in biology, reducing the chances of identifying the symmetries captured by either fibrations or automorphisms.

For instance, Chapters 15 and 16 have shown that graph fibrations uncover symmetries in gene regulatory networks that are validated by synchronization patterns in gene coexpression in Chapter 20. However, these results were shown for one particular organism, the bacterium *E. coli*, that is well-studied with a gene regulatory network and metabolism that has been almost completely mapped. In general, this is seldom the case. Structural networks are known only partially because only some pathways can be studied and mapped. This is particularly true for higher-level organisms like humans, where only certain gene interactions are studied. The inherent incompleteness and disordered nature of biological data preclude the application of the fibration formalism *as it is*. As a consequence, the algorithms to identify fibrations discussed so far may not uncover all the symmetries of the network.

Boldi et al. (2022) and Leifer et al. (2022) address this problem by proposing an extended theory of quasifibrations and pseudobalanced colorings, respectively (see Section 13.5). These theories attempt to capture more realistic patterns of almost-synchronization of units in biological networks.

Based on the theory of pseudo-balanced coloring, the network reconstruction algorithm developed by Leifer et al. (2022) (described in Section 13.5.3) aims to identify the optimal modifications needed for a network to achieve a symmetrically balanced coloring. This algorithm does not require prior knowledge of the balanced coloring. However, as we discuss next, it is often the case that the balanced coloring is known *a priori* through cluster synchronization. This information can be invaluable for reconstructing missing links in an incomplete network. Consequently, this algorithm is simpler and more practical than the alternative presented in Section 13.5.3 (Leifer et al., 2022).

Indeed, while determining complete structural networks is a challenge, obtaining complete functional networks for all genes in an organism can be done routinely. Current advances in systems biology make it possible to measure the activity of the entire transcriptome of a given species. For instance, the mRNA expressed by the ~20,000 human genes can be monitored simultaneously in cells using transcriptomic techniques like microarrays or RNA-seq, even at the single-cell level (Klipp et al., 2016). In the brain, single neuronal calcium activity can reveal whole-body synchronization in small connectomes, while fMRI can reveal similar synchronization at the mesoscopic level of larger brains. These patterns reveal the cluster synchronization emerging from the symmetries of the underlying structural network, as seen in Chapter 20.

We take advantage of these experimental advances to solve the inverse problem, from function \rightsquigarrow structure. Given an (almost) complete functional network from which cluster synchronization can be extracted, at least for a (large) set of nodes, we identify these synchrony clusters with the balanced colorings of the structural network. We are interested in inferring the underlying structural network supporting this synchrony (Fig. 22.1). Two main systems come to mind: (1) inferring the TRN from gene coexpression synchrony, and (2) inferring a connectome from neural synchronization, which we treat in Chapters 23, 24 and 25.

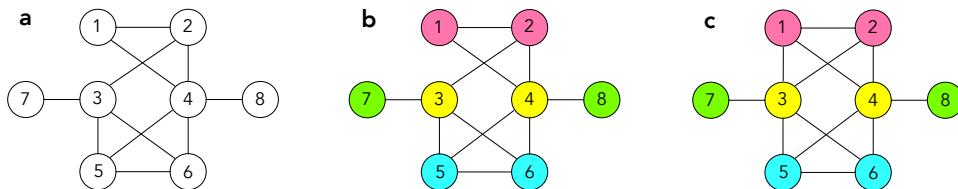


Fig. 22.1

Symmetry-driven inference algorithm informed by balanced coloring synchronization.

(a) Uncolored structured network. This network is incomplete. (b) A balanced coloring is obtained experimentally, measuring the cluster synchrony. It fails to be balanced at nodes 1 and 3. (c) Repairing the network by adding the minimal number of edges: an edge from 1 to 3 makes the coloring balanced. This example shows how a ‘missing’ link can break a balanced coloring. If we know the coloring in advance, it should be easy to infer this missing link. Thus, an optimization algorithm that looks for the minimal number of links to be added (or removed) from a network to satisfy a previously known balanced coloring would be an effective way to reconstruct the network. Such an algorithm makes use of fibration theory and balanced coloring to solve the following optimization problem: find the minimal number of changes to edges.

We wish to determine the missing links to complete these networks to fully characterize pathways responsible for gene expression and pathways in the brain. The existence of missing links and variability across samples preclude the identification of perfect symmetries in the connectivity structure. However, a symmetry-driven inference algorithm informed by cluster synchronization is able to reconstruct these missing links and remove erroneous links in the biological network.

As it turns out, the reconstruction algorithm will solve two problems that we treat in turn (Fig. 22.2):

- Link prediction (Fig. 22.2a): reconstructing an incomplete structural network (Section 22.2). We will apply this algorithm to reconstructing the connectome of *C. elegans* in Chapter 23.
- Pathway reconstruction (Fig. 22.2b): from a (quasi) complete structural network, infer the pathways that are active in a given task (Section 22.2.1). We will apply this algorithm to infer the pathways active in the memory engram in mice in Chapter 24 and in the human brain in Chapter 25.

22.2 Link prediction in biological networks

Modeling biological systems as networks with symmetries poses challenges. Firstly, experimental data on networks collected by different techniques are never complete due to experimental errors and the impossibility of measuring every possible link. Secondly, natural variability across individuals of the same species results in different networks making

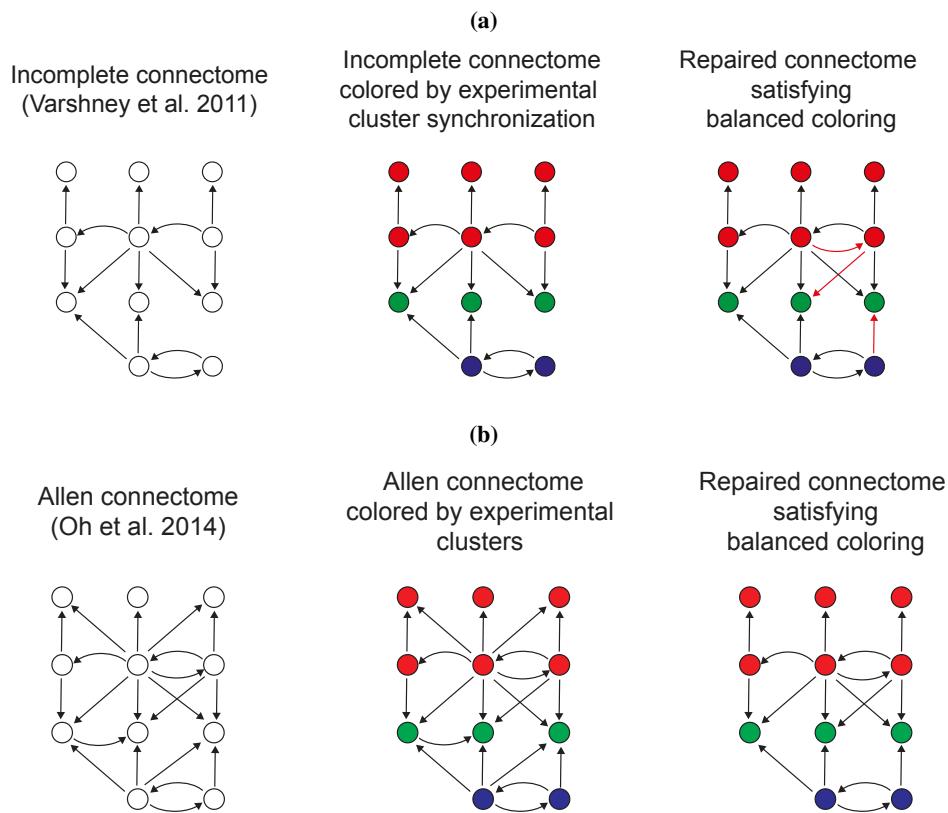


Fig. 22.2 Applications of symmetry-driven inference algorithm of structural networks. (a) Link prediction. Section 22.2 and Chapter 23. (b) Pathway reconstruction. Section 22.2.1 and Chapters 24, 25.

interpretations of the data across specimens more difficult. This is particularly important in the brain. The neuronal network of *C. elegans* contains 302 individually identifiable neurons, and the wiring diagram includes 890 gap junctions and 6,393 chemical synapses (White et al., 1986; Varshney et al., 2011). The number of neurons across these animals is very consistent (White et al., 1986), while the gap-junctions and chemical synapses are reproducible from animal to animal within 25% variability (Varshney et al., 2011; Hall and Russell, 1991). Due to its small size and relative completeness, the neural network of *C. elegans* has been a formidable model system in which to search for design principles underpinning the structural organization and functionality of neural networks.

Despite the fact that 302 neurons are very constant across worms, no two worms will ever have the same connectome. The concept of pseudo-balanced colorings arises naturally from the observation that connectomes vary from animal to animal. As just stated, Varshney et al. (2011) have estimated this variation experimentally to be 25% of the total connections. We consider this variability across individual connectomes to be an intrinsic property consistent

with biological diversity and evolution. Furthermore, the number of connections is subject to change from animal to animal through plasticity, learning, and memory, so it cannot be ignored.

On the other hand, while connectomes vary from animal to animal, functions developed from them, such as locomotion, are barely distinguishable between different worms, and still show some vestige of an ideal symmetry. In fact, the function is preserved despite the 25% variation in the connectomes. Despite this inherent disorder, the function and neural activity of *C. elegans* exhibits strong regularities, as observed, for instance, in its oscillatory locomotion patterns (Stephens et al., 2008) and the neural synchronization observed in neuronal activity (Kato et al., 2015).

This means, for instance, that a single balanced coloring in Fig. 22.2a center specifying a function could correspond to different slightly noisy connectomes for every animal shown on the left. Yet, upon the reconstruction of a few links, the ideal common average connectome can be revealed, shown on the right with the new links in red. Consequently, we expect deviations from exact symmetries to be relatively ‘small’. Exact symmetries of the connectome should be considered as an idealization, and we do not expect them to be realized exactly.

Thirdly, organisms adjust to a changing habitat using neural plasticity, epigenetics, and other adaptations. For example, even two organisms that are identical at a given moment may become different in the future. Organisms survive and benefit from these small variations, so evolution through natural selection occurs. Therefore, networks that model these systems exhibit not only differences across individuals but also have missing experimental links that can mask their underlying regularities. In particular, the missing experimental interactions and natural variations make the existence of perfect symmetries difficult to find in biological networks. This difficulty has persisted despite the ubiquitous existence of synchronization across all biological networks, which is a manifestation of (approximate) symmetries in the underlying networks that support biological activity.

What is the anatomical substrate for this synchrony? Figure 22.2a shows an example where, upon reconstruction of a few missing links, an incomplete network will reveal an underlying ideal perfect symmetry that is the unique ‘blueprint’ of the symmetry of the species. Each network can be thought of as a small variation on this perfect blueprint. The experimentally observed networks are all different and are always pseudosymmetric; they can be modified slightly to exhibit the ideal blueprint of the species.

Reconstructing a network to reveal ideal symmetries is an instance of link prediction problem in complex networks (Lü and Zhou, 2011). Link prediction is the problem of determining the probability of the existence of a link between a pair of nodes in the network that is missing from the data (Lü and Zhou, 2011; Getoor and Diehl, 2005; Liben-Nowell and Kleinberg, 2007; Chen et al., 2005; Clauset et al., 2008). It can also be generalized to removing spurious links from the network. Previous work has proposed several approaches to find missing links based on different statistical and Markov chain models; see the review in (Lü and Zhou, 2011). However, functional and dynamical features of incomplete networks, such as synchronization, are not taken into consideration in the reconstruction. The approach presented here employs network symmetry and synchronization as an organizing principle to guide link prediction and to reconstruct the network from its incomplete state.

22.2.1 Previous reconstruction algorithms for missing data

Existing approaches to network reconstruction (Brugere et al., 2018) infer TRN structures from coexpression based on pairwise correlations between the expression of genes; examples are ‘Weighted Gene Correlation Network Analysis’ (WGCNA) (Horvath, 2011; Zhang and Horvath, 2005; Langfelder and Horvath, 2008). Such approaches typically assume that shared expression profiles reflect control by the same TFs (Klipp et al., 2016; Horvath, 2011; Bansal et al., 2007; Brugere et al., 2018; Tegner et al., 2003; Liao et al., 2003; Wang and Huang, 2014). Reconstructing TRN direct interactions from pairwise correlations of gene expression in high-throughput data has been treated by many (Marbach et al., 2012; Bansal et al., 2007; Brugere et al., 2018; Butte and Kohane, 1999; Maertens et al., 2018; Chen et al., 2008).

Statistical methods for TRN reconstruction using expression data (Horvath, 2011; Zhang and Horvath, 2005; Marbach et al., 2012; Bansal et al., 2007; Brugere et al., 2018; Butte and Kohane, 1999; Maertens et al., 2018; Chen et al., 2008) typically do not distinguish between TRN and functional coexpression network (Roy et al., 2014; Klipp et al., 2016), that is, between the TRN and the network obtained typically by thresholding a coexpression matrix or correlation matrix.

This logic is also used to parametrize quantitative models (Kaplan et al., 2008; Liao et al., 2003; Klipp et al., 2016) and it underlies many attempts to infer regulation mechanisms from the observed expression profiles. However, as is clear from our analysis of, e.g., multilayer composite fibers in Section 15.4, this approach has problems since genes can show coexpression without sharing transcriptional inputs. They might just as well be regulated by different TFs that, by the symmetry of the network structure, tend to be coexpressed.

Thus, the difference between the functional coexpression network and the TRN is clearly demonstrated in multilayer composite fibers. In theory, this coregulation could be described by models, given the (usually unknown) activity profiles of different TFs. If the TRN structure implies such indirect coregulation, how can we infer this by network analysis alone? Comparing genes by their direct in-going links are not enough; instead, we need to go back and compare the entire chain of regulations behind these inputs. This process is achieved by the input tree and the symmetry fibration.

Basically, this implies that the typical approach to network reconstruction, which assumes that coexpression means direct co-regulation needs to be revised. It indicates the need to search for indirect interactions beyond co-regulated genes, including chains of regulations behind coexpressed genes.

22.2.2 Missing link algorithms using topological measures of the network

The link prediction problem is a problem of predicting the probability of the existence of an edge between two unconnected nodes, given the observed data. Typically, this probability is defined as the similarity between the nodes concerned, obtained using a topological measure of the graph (Clauset et al., 2008; Liben-Nowell and Kleinberg, 2007; Lü and Zhou, 2011). To repair a graph using a link prediction approach, an empirically chosen

number of top-ranked edges is predicted to exist (Chen et al., 2005). Common measures of similarity are divided into local, global, and quasi-local classes. We review some of them and propose a way to estimate their performance following Lü and Zhou (2011).

First, we introduce a few indices characterizing the local topology of the graph. Let S_{xy} be the distance between nodes x and y , let $\Gamma(x)$ be the neighborhood of node x (the set of nodes connected to x by a directed edge), let $|X|$ be the cardinality of set X , and let k_x be the degree of node x .

The *preferential attachment index* calculates the similarity between two nodes based on their degree:

$$S_{xy}^{PA} = k_x k_y. \quad (22.1)$$

This measure is widely used in scale-free networks, and is generalized by the *Randic index* (Randić, 1975; Kincaid and Phillips, 2011). In particular,

$$s(G) = \sum_{x,y \in V} S_{xy} \quad (22.2)$$

is called a *scale-free metric* and is used as a measure of scale-freeness of the graph (Li et al., 2005).

A degree-based metric is also used while generating a random scale-free network with the Barabasi–Albert model (Albert and Barabási, 2002). In this model, preferential attachment implies that each newly added node in a generated graph is connected to other nodes with a probability proportionate to their degrees. The scale-free metric (22.2) is used as a measure of scale-freeness of the graph.

The *Common Neighbors Similarity metric* is based on the assumption that nodes that have a lot of common neighbors are likely to be neighbors themselves. This assumption has had success in social networks (Newman, 2001; Kossinets, 2006); however, it does not increase the symmetry of the network. *Common Neighbors (CN)* is defined by:

$$S_{xy}^{CN} = |\Gamma(x) \cap \Gamma(y)|. \quad (22.3)$$

There are two differently normalized versions of the Common Neighbors metric: the *Salton Index* normalized by the degree of the nodes

$$S_{xy}^{Salton} = \frac{S_{xy}^{CN}}{\sqrt{k_x \times k_y}} = \frac{|\Gamma(x) \cap \Gamma(y)|}{\sqrt{k_x \times k_y}}, \quad (22.4)$$

and *Jaccard similarity* normalized by the total size of the node neighborhood:

$$S_{xy}^{Jaccard} = \frac{S_{xy}^{CN}}{|\Gamma(x) \cup \Gamma(y)|} = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|}. \quad (22.5)$$

A popular measure of similarity based on the global topology of the graph is the *Katz Index*. This index counts the number of paths between specified nodes (in the sense of this book; more technically, a ‘walk’, which can visit nodes and edges more than once) of increasing length, weighted by the coefficient $\beta < 1$. Here $(A)_{xy}^k$ is the number of paths of length k between nodes x and y (Newman, 2018). The Katz index is:

$$S_{xy}^{Katz} = \beta A_{xy} + \beta^2 (A)_{xy}^2 + \beta^3 (A)_{xy}^3 + \dots \quad (22.6)$$

This series converges when β is less than the inverse of the largest eigenvalue of A , and the sum is

$$S_{xy}^{Katz} = ((I - \beta A)^{-1} - I)_{xy}, \quad (22.7)$$

where I is the appropriate identity matrix.

These methods are used in typical missing link studies. Since all methods are heuristic, the accuracy of each method needs to be assessed with traditional hypothesis testing performance metrics. The confusion matrix in Table 22.1 introduces four basic metrics: true positive (TP), false negative (FN), false positive (FP) and true negative (TN), which compares edges identified by the manual ‘ground truth’ solution (true/false) with the edges obtained by a method (positive/negative). Other important metrics include Precision, Recall, Miss Rate, Accuracy, and F-measure, which are calculated as functions of TP, FN, FP, and TN as:

$$\begin{aligned} \text{Precision} &= \frac{TP}{TP + FP}, \\ \text{Recall} &= \frac{TP}{TP + FN}, \\ \text{MissRate} &= \frac{FN}{FN + TP}, \\ \text{Accuracy} &= \frac{TP + TN}{TP + FP + FN + TN}, \\ \text{F - measure} &= \frac{TP}{TP + \frac{1}{2}(FP + FN)}. \end{aligned} \quad (22.8)$$

Using the above methods, we can choose the edges to be repaired with the highest priority. When the network is sparse, the TN value is very high for all methods. Therefore, the best metric for overall performance assessment is the F-measure, which is proportional to the number of edges guessed correctly and inversely proportional to the sum of missed and falsely identified edges.

22.3 Inferring the pathways guided by cluster synchronization

A second application of the reconstruction algorithm is to determine the pathways in the structure network associated with a given synchronization task that produces a balanced coloring in the network, see Fig. 22.2b. Here, it is assumed that the connectome is well-known.

The main application is in the brain, although the method can be applied to any network. It is based on an allegory of a highway network and the one-to-many degeneracy into many functions elaborated by Park and Friston (2013). A baseline connectome serves as a

		Prediction by a method	
		Predicted	Not predicted
Ground truth	Predicted	True Positive (TP). Correct prediction. Edge repaired by ground truth and a method	False Negative (FN). Type II error. Edge repaired by a ground truth, but not repaired by a method
	Not predicted	False Positive (FP). Type I error Edge repaired by a method, but not repaired in ground truth	True Negative (TN). Correct rejection. Edge not repaired by either ground truth or a method

Table 22.1: **Confusion matrix.** The confusion matrix identifies four variables, TP, TN, FP, and FN, depending on the positive or the negative outcome of the predicted result and the ground truth. The comparison assumes that we know the ground truth. For instance, we start with a perfectly symmetric network and then remove a few links. Then, we test the methods to find these missing links. The ground truth is known since we removed the links. True and false correspond to the ground truth and positive and negative correspond to the prediction.

detailed map of the intricate highway system that underlies the brain's various information pathways. This is illustrated in Fig. 22.2b left by the Allen connectome for the mouse (see Chapter 24). These pathways are the substrate for all functional tasks that the brain performs. It is important to note that the specific routes along this neuroanatomical highway vary depending on the nature of the task at hand. According to the hypothesis proposed by Park and Friston (2013), the brain's functional activity selectively engages a subset of the available connections within this expansive 'highway' connectome. This selective engagement allows the brain to adaptively operate in diverse functional states, whether at rest, during memory retrieval, or while executing motor functions. The phenomenon described as a 'one-to-many' degenerate structure-function relationship highlights how a single, static architecture of neural connections can give rise to a multitude of distinct functional states, showcasing the remarkable flexibility and complexity of the brain's operations.

Thus, active pathways in the brain adjust according to the task at hand (Friston, 2011; Park and Friston, 2013). Algorithmically, starting with a baseline structural network (shown on the left of Fig. 22.2b) and a balanced coloring of the network (depicted in the center, which is obtained from a specific task), the algorithm will identify the minimal number of removals (and possibly additions) to the connectome. This process aims to discover a substrate network that reproduces the balanced coloring, as illustrated on the left in the figure. This algorithm will be applied in Chapter 24 to the mouse brain and Chapter 25 to the human brain.

22.4 Link prediction in biological networks via symmetry and synchronization

We now develop the algorithm to infer the structural network from a balanced coloring of the functional network. Ávila et al. (2025a,b); Gili et al. (2025) use the pseudobalanced

coloring theory of (Leifer et al., 2022) to develop a mixed integer linear programming (MILP) (Bertsimas and Tsitsiklis, 1997) algorithm to optimize a minimal link removal (and addition) from a baseline structural network to satisfy a predefined balanced coloring of the network.

The inference algorithm can be summarized in the following steps:

- First, identify the baseline structural network (left of Fig. 22.2a or b) that forms the graph of all known structural connections between the nodes. This is the baseline structural network that will be repaired by the algorithm.
- Using any synchronization measure, find the functional network of the system from experimental activity data. This can be the gene coexpression matrix or synchronous neural data. Extract the cluster synchronization of the nodes from the functional network. Assign a color to each cluster. Assign to each node in the baseline structural network a color according to the cluster to which they belong (center of Fig. 22.2a or b). This is the balanced coloring that guides the network reconstruction.
- Apply the MILP algorithm to reconstruct the baseline network by adding/removing the minimal number of edges until the balanced coloring of the decimated graph matches the coloring obtained from the functional network (right of Fig. 22.2a or b). According to the application, adding, removing, or both should be applied.

22.5 Integer linear program to infer the structural network from cluster synchronization

The crucial step in this scheme is finding minimal additions/removals that satisfy the coloring condition. This problem is formulated as a mixed integer program that is solvable for modestly sized instances. Ávila et al. (2025a) formulate the problem of finding the minimum perturbations to induce a minimal balanced coloring as an integer linear program, i.e., an optimization problem where the decision variables are all integers. Ávila et al. (2025b) have applied this algorithm to reconstruct the *C. elegans* connectome (Chapter 23). García-Hernández et al. (2025) used it to unravel the minimal engram structure during memory formation in mice (Chapter 24), and Gili et al. (2025) applied this algorithm to the human brain (Chapter 25). The most updated code appears in GitHub.com/MakseLab.

We consider a directed graph, $G = (V, E)$, where V denotes the set of nodes, and E denotes the set of directed edges (recall that an undirected edge is considered as two directed edges). Let $n = |V|$ be the number of nodes and let $m = |E|$ be the number of directed edges. We also define

$$E^C = \{ij : i, j \in V, ij \notin E\} \quad (22.9)$$

as the set of ordered pairs of nodes for which no directed edge exists in G , which we refer to as *non-edges*. These ordered pairs represent potential edges that could be added to the graph G . We define α, β as constant parameters that govern the objective function's relative importance between edge removal and edge addition.

Definition 22.1 Symmetry-driven inference algorithm guided by balanced coloring synchronization. Let \mathcal{S} denote a coloring of G . This coloring is provided by the cluster synchronies from the functional network synchronization. We wish to determine the minimum number of edges to add or remove so that \mathcal{S} becomes a balanced coloring of G .

The following integer programs are guaranteed to find a balanced coloring, but are not guaranteed to find a minimal balanced coloring.

The model's three families of binary decision variables are defined as follows.

For $ij \in E$,

$$r_{ij} = 1 \text{ if edge } ij \text{ is removed, 0 otherwise.} \quad (22.10)$$

For $ij \in E^C$,

$$a_{ij} = 1 \text{ if non-edge } ij \text{ is added, 0 otherwise.} \quad (22.11)$$

For $P, Q, R \in \mathcal{S}$ with $P \neq Q$ and for $i \in P, j \in Q$

$$s_{ijR} = 1 \text{ if } i \text{ and } j \text{ are imbalanced on } R, 0 \text{ otherwise.} \quad (22.12)$$

The role of the linear constraints below is to set up a system of linear equalities and inequalities that, if satisfied by these decision variables cause the given coloring to be a balanced coloring of the resulting repaired graph.

The objective function is to minimize the weighted sum of edges removed and edges added. The function is then defined as:

$$f_{\alpha,\beta}(r, a) = \alpha \sum_{ij \in E} r_{ij} + \beta \sum_{ij \in E^C} a_{ij}. \quad (22.13)$$

This equation represents a simplified version of unweighted networks (which include multi-graphs if we replace positive integer weights by edges of suitable multiplicity). Below, we generalize the approach to include weighted networks. The main constraint ensures that \mathcal{S} is a balanced coloring of the perturbed graph G .

$$\begin{aligned} & \sum_{ip \in E: i \in S} (1 - r_{ip}) + \sum_{ip \in E^C: i \in S} a_{ip} = \\ & \sum_{iq \in E: i \in S} (1 - r_{iq}) + \sum_{iq \in E^C: i \in S} a_{iq}, \quad p, q \in T; S, T \in \mathcal{S}. \end{aligned} \quad (22.14)$$

Constraints (22.14) are imposed for every pair of nodes p, q that have the same color and for every color set. for a given edge $ij \in E$, the quantity $1 - r_{ij}$ is 1 if the edge is not removed and 0 if it is removed. Also, for $ij \in E^C$, the quantity a_{ij} is 1 if ij is a newly created edge and 0 otherwise. Thus, the left-hand side of (22.14) represents the edges that enter into a given node p from the color set S , and the right-hand side represents the edges entering node q from S . Using the same sums, (22.15) ensure that the in-degree is at least 1 for every node:

$$\sum_{ip \in E} (1 - r_{ip}) + \sum_{ip} a_{ip} \geq 1, \quad p \in V. \quad (22.15)$$

The following constraints are valid for minimal balanced colorings; i.e., they are necessary but not sufficient.

$$\begin{aligned} & \sum_{ip \in E: i \in R} (1 - r_{ip}) + \sum_{ip: i \in R} a_{ip} - \left(\sum_{iq \in E: i \in R} (1 - r_{iq}) + \sum_{iq: i \in R} a_{iq} \right) \\ & \geq s_{pqR} - ns_{qpR}; \quad p \in S; q \in T; R, S \neq T \in \mathcal{S}, \end{aligned} \quad (22.16)$$

$$\begin{aligned} & \sum_{iq \in E: i \in R} (1 - r_{iq}) + \sum_{iq: i \in R} a_{iq} - \left(\sum_{ip \in E: i \in R} (1 - r_{ip}) + \sum_{ip: i \in R} a_{ip} \right) \\ & \geq s_{qpR} - ns_{pqR}; \quad p \in S; q \in T; R, S \neq T \in \mathcal{S}, \end{aligned} \quad (22.17)$$

$$s_{pqR} + s_{qpR} \leq 1; \quad p \in S; q \in T; R, S \neq T \in \mathcal{S}; \quad (22.18)$$

$$\sum_{R \in \mathcal{S}} (s_{pqR} + s_{qpR}) \geq 1; \quad p \in S; q \in T; S, T \in \mathcal{S}. \quad (22.19)$$

The inequalities (22.18) keep at most one of the two binary variables s_{pqR} equal to 1 for every color R . If both are zero, then the inequalities (22.16) and (22.17) force p and q to be balanced for the color R . If one is zero, the total in-adjacent nodes of color R for p and q differ by at least 1. In particular, for color R , if $s_{pqR} = 1$ and $s_{qpR} = 0$, then the number of in-adjacent nodes to p is at least 1 greater than that to color q . The converse is also true.

The inequalities (22.19) force one of s_{pqR} or s_{qpR} to equal 1 for at least one color R . This is a necessary but not sufficient condition for the coloring to be *minimal*. For example, if two color partitions have no edges between them, the same number of edges to all other colors, and the same positive number of internal edges, then (22.19) is satisfied since different colors would register as an imbalance. However, the union of these two color partitions is balanced and has one color fewer; i.e., the coloring is no longer minimal. That being said, experiments by Gili et al. (2025) yield strong evidence that in practice, the necessary condition enforces the minimal balanced condition, since a minimal balanced coloring was found for all test cases.

The complete model is then:

$$\begin{aligned} & \min f_{\alpha, \beta}(r, a) \\ & \text{subject to} \quad (22.14), (22.15), (22.17), (22.17), (22.18), (22.19), \\ & \quad r_{ij}, a_{k\ell}, s_{pqR} \in \{0, 1\}, ij \in E, \\ & \quad k\ell \in E^C, p \in P, q \in Q, P \neq Q, R \in \mathcal{S}, \end{aligned} \quad (22.20)$$

where (22.19) within the equation above is a reference to select only one of its sub-equations.

The objective and constraint functions are linear. We then solve the integer linear programs with the solver Gurobi Optimization, LLC (2023). Ávila et al. (2025a) have benchmarked and validated this algorithm. The uniqueness of the solution is tested by developing an independent solver based on the quasifibration framework of (Boldi et al., 2022). In all cases considered, we find the same solution using the quasifibration formalism and MILP.

The drawback of this technique is that MILPs are, in general, NP-Hard. However, decades of research from the 1990s and computational advances have increased our ability to solve

MILPs by several orders of magnitude. Since 2000, advances in hardware alone have increased the speed by a factor of 180 to 1000, and algorithmic progress has rendered many previously intractable problem classes solvable in seconds (Koch et al., 2022). Future directions are to investigate algorithmic improvements to solve larger instances.

The weighted case requires new variables and a modification of the constraints (22.14). Let w_{ij} denote the weight of edge ij for $ij \in E$. Let $\varphi \in [0, 1]$ denote the maximum percentage by which a preexisting edge weight can be adjusted, and let τ denotes the maximum weight that a new edge can possess. Let μ_{ij} denote the modification of weights for preexisting weights for $ij \in E$. Let σ_{ij} denote the weight of a new edge if one is added for $ij \in E^C$.

The main constraint to ensure that S is a balanced coloring of the weighted graph, G is:

$$\begin{aligned} & \sum_{ip \in E: i \in S} (w_{ip}(1 - r_{ip}) + \mu_{ip}) + \sum_{ip: i \in S} \sigma_{ip} \\ &= \sum_{iq \in E: i \in S} (w_{iq}(1 - r_{iq}) + \mu_{iq}) + \sum_{iq: i \in S} \sigma_{iq}, \quad p, q \in T, S, T \in \mathcal{S}. \end{aligned} \quad (22.21)$$

The following constraints ensure that the new weight variables are restricted to a percentage of the current weight or an absolute number for edge weight modifications or new edge weights, respectively: $-\varphi w_{ij} \leq \mu_{ij} \leq \varphi w_{ij}$.

As mentioned, the algorithm is not just about repairing missing links, but identifying the routes that sustain the functional networks in different dynamical states. Even within a single connectome, the brain can never have exact symmetries due to its high complexity. Structural brains are all different, but a certain level of ideal symmetry must be common to all of them in order to guarantee the performance of the same function, even though not all structural brains are identical. We elaborate on these applications in the next three chapters.

Fibration Theory of the Brain I: *C. elegans* Locomotion

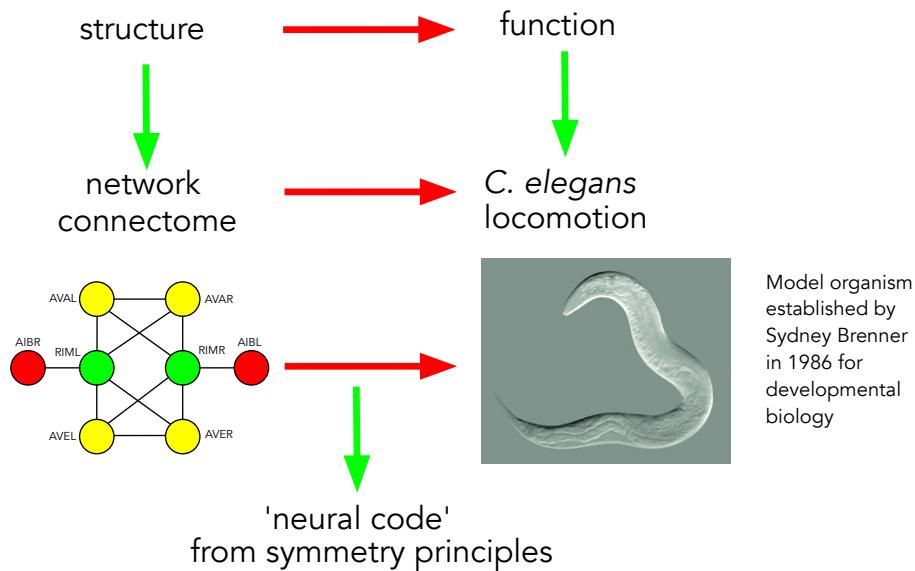
Physics hopes for a simplified vision of nature, and this wisdom was clearly realized in the fiber bundle picture of the fundamental forces of nature (Chapter 10). The hope for a simplified vision in biology encounters its biggest challenge in the search for an organizing theory of the brain. The brain works by processing signals traveling mainly through the connectome of synaptic connections and the information flow in these networks is related to the fibration; a rigorous extension of the fiber bundle. The question arises: Can a theory based on fibration symmetries help uncover the organizing principles of the brain by shedding light on the structure of the connectome and its relation to function? This chapter starts by tackling this question for one of the simplest neural systems mapped as of today: the 302-neuron system of the nematode *C. elegans*. We discuss synchronization in the organism's locomotion connectome. Succeeding chapters treat similar questions for rodents and the human brain.

23.1 Can we hope for an organizing theory of the brain?

The central hypothesis of the present book applied to the brain, is that the structure of the connectome—that is, the pattern of interconnections between neurons—is not random, and the regularities observed in the structure of connectomes can be explained by symmetries of the biological network, which is then manifested in neuronal synchrony and function. This hypothesis is exemplified in Fig. 23.1. It addresses a core problem in systems neuroscience (Kandel et al., 2013; Yuste and Church, 2014; Alivisatos et al., 2013; Hartwell et al., 1999b; Tononi et al., 1994): how the function of the brain emerges from the structure of the connectome through the neural code. Searching for such organizing principles in a biological system is a goal of systems science in general, but this question becomes particularly difficult for the brain because of the enormous complexity of the brain connectivity.

Typically, the structure of connectomes has been modeled by random models (Bassett and Bullmore, 2006; van den Heuvel and Sporns, 2013; Watts and Strogatz, 1998; Barabási, 2009). A typical configuration of a random model resembles Fig. 23.2 left, which depicts the entire *C. elegans* connectome. Features of randomness, such as the scale-free nature (Barabási, 2009), are suggested by the broad range of sizes of the nodes (which are plotted in proportion to the degree of the node), and the random disorder of a small-world network (Watts and Strogatz, 1998), are apparent in the long-range links.

In contrast, we can plot the same network by placing each node in a two-dimensional

**Fig. 23.1**

Structure ⇔ function relation in the nematode connectome. The structure is characterized by its symmetries, which determine the patterns of locomotion of the phenotype.

plot that respects its symmetries. This plot results in the ordered structure on the right of Fig. 23.2. This figure depicts the same connectome as that on the left but with the neurons in harmony with their symmetries. While this is just a pictorial representation, it makes the point that an organizing principle for the brain could extract order from the biological disorder inherent in small-world and scale-free random models of the brain.

In Sections 12.6.2 and 18.3 we developed the idea that gene duplication provides a plausible mechanism for the emergence of symmetry structures in genetic networks. In Section 21.7, we also speculated on whether symmetries are a consequence of biological evolution. Evolution, with its strong selective pressure toward coherent structures, might be sufficient to produce connectomes with symmetries, structured hubs, and small-world networks, all at the same time.

Similarly, we can argue that synchronized fibers in biology cannot be discovered only through statistical comparisons with random models. Fibers determined by symmetries are not network motifs. Important symmetry structures might appear only once in the network and would not then be statistically over-represented. In the tradition of physics, synchronized fibers are uncovered only with guidance from theory, not through statistical analysis. Fibers may not be over-represented, as network motifs are (by definition), but they are significant for function.

Since symmetries require strong structural patterns of connections, which can easily be broken by a single missing or extra link, we wonder whether symmetries can survive the natural variability and complexity of the brain. Indeed, even for the simplest of all connec-

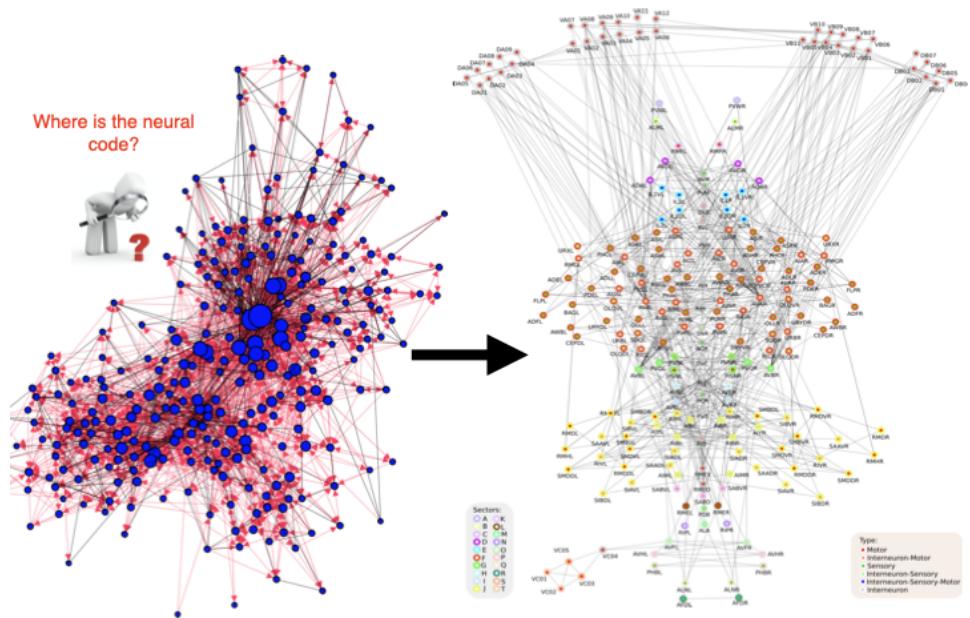


Fig. 23.2

Where on Earth is the neural code? Symmetries bring order out of biological disorder. *Left:* The worm connectome, as represented in typical plotting software like Gephi, shows pictorially random characteristics like a small-world and scale-free dominated by hubs. Node size is proportional to the degree. Where is the neural code represented in such a connectome? *Right:* For comparison, this plot represents the organization of the same connectome by its symmetries. This organization agrees in broad terms with the classification into classes: interneurons, sensory and motor neurons, and their combinations. Might this figure be a more faithful representation of the structure of brains than random models?

tomes, that of *C. elegans*, no two connectomes of different worms are ever exactly the same. As discussed in Section 22.2, there is a natural variability of around 25% between connectomes of different worms, and this variability is sufficient to break all symmetries. Thus, to capture symmetries in real connectomes, with missing links and variability from animal to animal, we introduced the concepts of pseudosymmetry and quasifibration in Section 13.5, and in Chapter 22 we presented algorithmic versions that can reconstruct imperfect connectomes and realize their inherent underlying ideal symmetries. These ‘almost-symmetries’ can capture natural differences between connectomes of organisms in the same species, and can tell us how far away from ideal symmetry, the connectome of a given worm is.

Even for the smallest known connectome it is already difficult to find symmetries, as evidenced by the need to use complex NP-hard algorithms for pseudosymmetries and

quasifibrations. We might, therefore, ask whether it is hopeless to search for symmetries in more complex brains, and in particular, the human brain. As soon as we consider other model organisms beyond *C. elegans*, the complexity of brain connectivity becomes intractable. The sizes of known connectomes of model organisms span 11 orders of magnitude from the 302 neurons of *C. elegans* (White et al., 1986) to the 100,000 neurons of the larval zebrafish (Ahrens et al., 2013) and the adult fruit fly (Zheng et al., 2018), the 70 million of the house rat, the 6.4 billion of the rhesus macaque, and ending with the 86 billion neurons in the human brain, which is, alas, not the largest in the animal kingdom. Certainly, we cannot expect such complex connectomes to respect the strict rules leading to exact symmetries.

In this chapter and the next two, we explore the symmetries of connectomes across all these scales and species. We devote some time to *C. elegans*, showing that the entire connectome at the single-cell level can be ordered in terms of symmetries. We then analyse more complex species, rodent, and human brains at a coarse-grained level with known partially reconstructed mesoscopic connectomes and exhibit further symmetries in those. In these more complex connectomes, our approach is to use the reconstruction algorithms of Chapter 22 to invert the arrow from function \rightsquigarrow structure, and to infer a meaningful mesoscale connectome that provides predictions for the structure and response function of the brain. These predictions are then validated experimentally.

We argue that synchronization is widespread in these systems, so symmetry at a given scale (perhaps not the cellular scale, but larger axonal fiber scales) should support these coherent states. We analyze in detail synchronization of groups of the order of thousands of neurons, as measured by c-Fos activity in rodents under a memory consolidation tasks and in fMRI signals obtained in humans performing language-related tasks. We argue that symmetries in the underlying network connectomes of large-scale axonal fibers could support brain synchronization at large scales in the human brain. Network symmetries lead directly to the ensemble of neurons that synchronize their activity and can thus be associated with an ensemble of functionally related neurons.

A note on nomenclature: to avoid confusion, we henceforth refer to the anatomical connections between mesoscopic areas in the human brain across white matter as ‘axonal fibers’ or ‘white matter tracts’. We reserve the term ‘fiber’ for a fiber of a fibration.

23.2 Dynamics and structure \rightsquigarrow function in the brain

The Hebbian learning rule (Hebb, 1949) has been informally characterized as:

Box 23.1

Hebbian rule (Hebb, 1949)

‘cells that fire together wire together’

This is not quite what Hebb intended; his idea was that if one cell should *cause* another

to fire, the strength of the connection between them should increase. Thus ‘fire together’ should not be interpreted as synchrony.

Our basic principle is similar but differs in one vital respect: a *direct* connection between the cells is not required. It is:

Box 23.2
Our hypothesis

‘cells that fire together have symmetries together’

If a connectome possesses symmetries, we expect the symmetries to be reflected in the neural synchronization and then in the functionality of the corresponding circuit. Moreover, we expect symmetry measures to outperform other popular connectivity measures when identifying functional building blocks.

Finding organizing principles for the wiring of the brain from symmetries also lead to interesting analogies with the wiring of artificial devices, and in Chapter 26 we discuss how this information might inspire a new generation of AI architectures-based on fibrations.

The theoretical framework of symmetries does not directly take into account the dynamical system of equations describing neuronal activity. To be precise, it assumes that the structure of the connectome can determine the functional classification of the neurons independently of the dynamics. The question arises: Can the dynamics break symmetry, even when the network topology is symmetric?

The answer is ‘yes’. Just as solutions to nonlinear differential equations can spontaneously break the symmetry of these equations; it is possible for certain dynamic solutions to a network ODE to break (at least some) symmetries of the network. Such behavior is the *synchrony breaking* discussed in Chapter 8. Recall that we assume that the dynamics are specified by a set of admissible equations (Definition 3.10)—equations that respect the fibration symmetry of the network. The existence of symmetries imposes a hard constraint on the dynamics, and the induced synchronization is largely independent of the type of dynamical equation used to describe the system—as long as the equations are admissible.

However, the dynamics could break the symmetries imposed by the structure. Imagine, bursting neurons following a Hodgkin–Huxley model (Izhikevich, 2003) or similar. There could be a set of parameters and initial conditions leading to solutions that break the symmetries of the network. Indeed, if the fully symmetric state becomes unstable as some parameter varies, the system can (and usually will) bifurcate to a less symmetric state. This causes synchronized clusters to break into smaller ones. Section 8.8.3 illustrates such possibilities for the metabolator and Smolen networks, with a mere two nodes. Even in this simple case, there are many different combinations of stable states.

Moreover, the dynamics of neurons are not only regulated by interactions via the connectome but also by other interactions, most notably by neuromodulators, which can regulate a population of neurons (Bargmann, 1993); some of them are discussed in terms of the multiplex models of Section 9.5. These interactions could induce a plausible breaking of symmetry that is not taken into account by the network symmetry. What part of the functionality of the neural circuit can be captured by the structure of the connectome itself? Results

below indicate that classifications, such as command interneurons in different locomotions and motor neurons are related to the building blocks of *C. elegans*. This approach explores how much of the functionality of the connectome can be captured by its network topology.

23.3 The *C. elegans* neural system

The nematode *C. elegans* is a primitive organism, over 600 million years old, with a simple neural system made of a network of gap junctions and another network of chemical synapses. This connectome has been studied over the last four decades and fully mapped, neuron by neuron and axon by axon, using electron microscopy to scan every single synaptic connection. The smallness and completeness of this connectome make it a perfect system to start searching for symmetries in neural networks.

The *C. elegans* connectome is paradigmatic in neuroscience. *C. elegans* was first established as a model organism in genetics by Sydney Brenner in 1963 to study developmental biology (Brenner, 1974). By 1986, its connectome was mapped by White et al. (1986). Today it remains the only connectome reconstructed in its entirety. Despite it being the simplest of the known connectomes and having been fully mapped four decades ago, we still do not know how to simulate the mind of the worm. We do not even know how to simulate in a computer its elegant sinusoidal locomotion (reflected by its name) using a model based on the relevant part of the connectome, although principal component analysis yields a simple dynamical system based on ‘eigenworms’ (Stephens et al., 2008). A fair question is whether a complete knowledge of the synapse-level wiring diagram of a brain, as is available for the worm, is enough to understand the neural code, and, from there, to predict motion, function, and behavior. The search for an organizing principle for the brain is wide open.

Fibration symmetry allows the study of building blocks of connectomes and their relation to synchronization and function. The symmetry hypothesis can be tested rigorously in *C. elegans* using the synapse-resolution connectome in conjunction with dynamical data from the nervous system at single-cell resolution. This approach can then be extended to analyze available connectomes across more complex species, which are yet not complete. However, partial reconstructions and dynamical data exist, so the symmetry hypothesis can be tested across numerous species, from annelids, fruit flies, zebrafish, rodents, macaque, to humans. These results in turn, can be used to develop a unifying symmetry theory of the brain to uncover synchronized neural populations that could be functionally related at different scales, from the single cell in *C. elegans* to the mesoscopic scales of the human brain captured by fMRI activation.

As stated above, the neuronal network of *C. elegans* is made of 302 neurons, which are individually identifiable. Its wiring diagram includes 890 gap junctions and 6,393 chemical synapses, according to the reconstruction curated by Varshney et al. (2011). Because of its small size and relative completeness, the neural network of *C. elegans* has been a formidable model system in the search for design principles underpinning the structural organization and functionality of neural networks (White et al., 1986; Chalfie et al., 1985; Bargmann,

1993; Gray et al., 2005; Chen et al., 2006; Varshney et al., 2011; Kato et al., 2015; Yan et al., 2017).

Symmetries of connectomes are either permutations of neurons in the neural circuit (orbits) or fibration symmetries of the input trees that define the synchronized fibers. Morone and Makse (2019) analyze the symmetries of the *C. elegans* connectome by looking at the circuits involved in the forward and backward locomotion tasks. Backward locomotion of the animal is supported by the activation of neural classes of interneurons AVA, AVE, AVD, AIB and RIM, and two classes of motor neurons, VA and DA (ventral and dorsal motor neurons of class A). Similarly, forward locomotion is supported by the activation of motor neuron classes VB and DB through the interneuron classes AVB, PVC, and RIB. These neurons are used, together with the connectome of gap-junctions and chemical synapses from (Varshney et al., 2011), to construct the neural circuits for forward and backward locomotion.

Figure 23.3 shows a hand-crafted reconstruction of the forward locomotion gap-junction circuit in *C. elegans* from (Morone and Makse, 2019). It shows that the symmetries of this locomotion circuit are determined by automorphisms. The symmetry group can be factorized into a direct product of normal subgroups, as shown in the figure:

$$\mathbf{F}_{\text{gap}} = \mathbf{C}_2 \times \mathbf{C}_2 \times \mathbf{S}_5 \times \mathbf{D}_1 \times \mathbf{C}_2 \times \mathbf{C}_2. \quad (23.1)$$

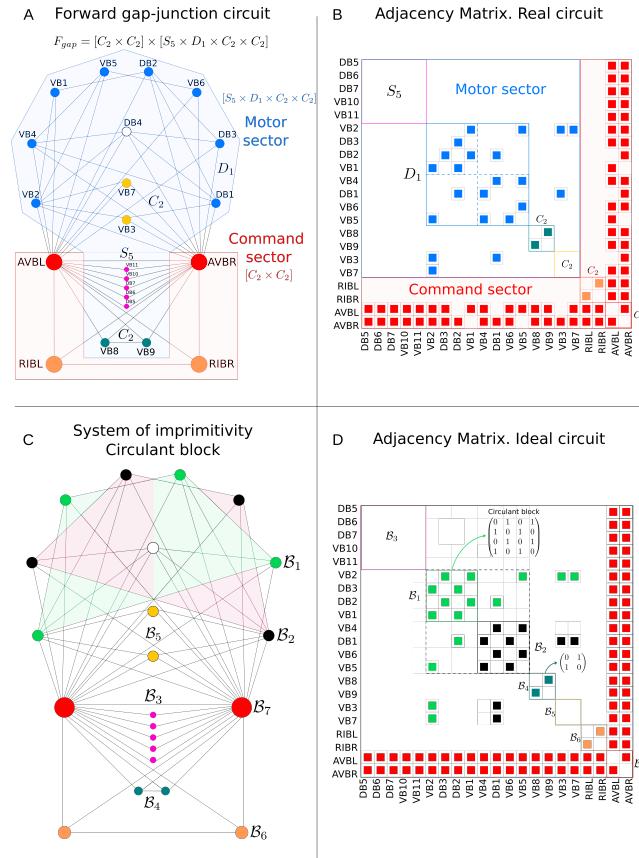
We have already explained that automorphisms are more restricted symmetries than fibrations and are less common in biology. Despite this, automorphisms appear in the gap-junction connectome. They may be present because the *C. elegans* gap-junction circuit is quite elementary as an evolving brain. However, we have not found any other biological network described by automorphisms. More complex organisms exhibit almost no automorphisms; most of their symmetries are fibrations.

Automorphisms constitute a special class of fibration symmetries: the orbits are the fibers, and the fibration is a projection from orbits to a set of representatives. Gap-junctions are undirected links, but the chemical synapse network is made from directed links and also has many automorphisms (Morone and Makse, 2019). Again, this may be due to the simplicity of the neural system. However, in a later study, Ávila et al. (2025b) also found extra fibration symmetries not captured by automorphisms. So, in this network, we find both orbits (of the symmetry group of a subnetwork) and fibers that are not orbits of any such group. We discuss this more general framework in succeeding sections.

23.4 Synchronization in *C. elegans* neural system

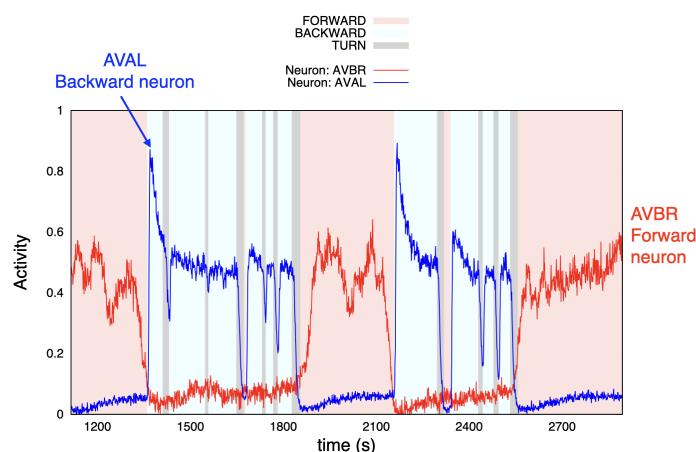
Experimental evidence for synchronization in brain activity is abundant and has been observed at all scales, from single neurons to large-scale EEG waves and fMRI activity. There are excellent books and reviews that elaborate on all these examples (Strogatz, 2018; Arenas et al., 2008; Pikovsky et al., 2001).

Evidence of synchronization in neural populations in *C. elegans* can be obtained by probing brain dynamics at single-cell resolution and in real-time, i.e., sub-second scale.

**Fig. 23.3**

Symmetry group F_{gap} of the forward gap-junction circuit of *C. elegans*. (a) Real circuit with connections from (Varshney et al., 2011) showing the normal subgroup factorization and its match with the broad classification of command interneurons and motor neurons. This circuit has pseudosymmetries since it is not complete. (b) The adjacency matrix of (a) shows the subgroup structure and its matching with broad neuronal classes. The incompleteness of the dataset is apparent in the matrix. (c) Ideal circuit of (a). This circuit is a hand-crafted reconstruction of the original circuit in (a) obtained by adding the smallest set of edges to obtain symmetry below the variability threshold of 25%. An internal structure made of blocks of imprimitivity (Morone and Makse, 2019) is visible in the cycles shown in the circuit. (d) Ideal adjacency matrix showing how the subgroups are broken down into blocks of imprimitivity. Figure reproduced from (Morone and Makse, 2019). Copyright © 2019, The Author(s).

Many experimental groups are working on full-body single-cell resolution. Below, we review the work of Manuel Zimmer and collaborators at the University of Vienna (Kato

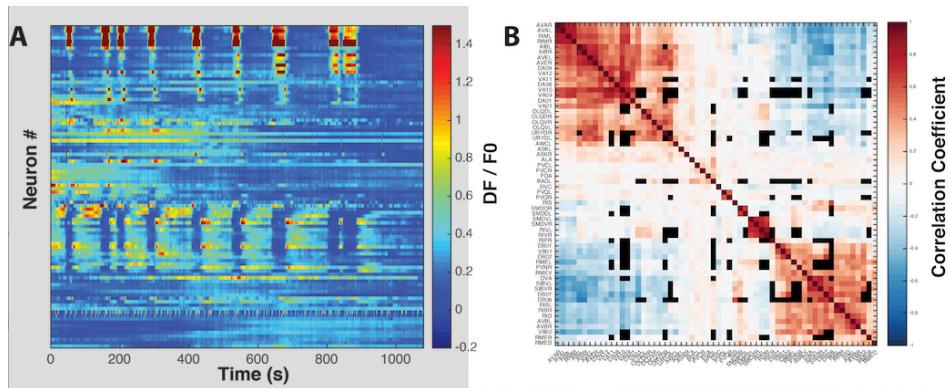
**Fig. 23.4**

Neuronal activity at single-cell resolution. Calcium image activity of two neurons, AVAL and AVBR, involved in forward and backward *C. elegans* locomotion, respectively, are shown by their anticorrelated behavior. Data collected by the Zimmer lab.

et al., 2015; Nichols et al., 2017; Skora et al., 2018), which is particularly suitable for testing the relation between symmetries and synchronization under locomotion; see also (Nguyen et al., 2016). These groups have pioneered experiments using Ca++ imaging data that encompass the whole nervous system, including head and tail ganglia as well as the ventral nerve cord containing all motor neurons. This permits a study of the relationship between synchronization and network structure for the particular function of locomotion, which has also been extensively studied before (Bargmann, 1993).

Data have been generated under various experimental conditions, including variation in feeding status, developmental state, sensory stimulation, and genotypes (Schrödel et al., 2013; Kato et al., 2015; Nichols et al., 2017; Skora et al., 2018). Results for real-time and single-cell resolution whole-brain calcium imaging in *C. elegans* have been published in (Schrödel et al., 2013), and subsequent descriptions of internal brain dynamics that correspond to behavioral action commands, as well as arousal and sleep states, are published in (Kato et al., 2015; Nichols et al., 2017; Skora et al., 2018). These dynamics involve a large proportion of neurons in the animals' brains, providing sufficiently large datasets to analyze thousands of potential neuronal pairwise interactions.

The functional imaging assays permit a first-level analysis of synchronization in the connectome. In these experiments, transgenic worms expressing a nuclear-localized Ca++ -sensor (GCaMP) in a pan-neuronal fashion are immobilized in microfluidic devices (Schrödel et al., 2013; Kato et al., 2015). This enables volumetric fluorescence imaging at sufficient speed and optical resolution to capture the activity of most neurons with single-cell resolution in the animal's head ganglia (Schrödel et al., 2013; Kato et al., 2015).

**Fig. 23.5**

Synchronization in worm locomotion in the experiments of Kato et al. (2015). (a) Heat map of whole brain Ca++ imaging data of brain activity in a typical worm. (b) Matrix of pairwise correlations between the activities of activated neurons under locomotion. Spontaneous activity is dominated by synchronies as shown in the heat map. The correlation matrix shows the Pearson coefficient between the neurons, which reveals clusters of synchronized neurons. Figure reproduced from (Kato et al., 2015). Copyright © 2015, Elsevier Inc.

Figure 23.4 shows an example of calcium image activity in two neurons involved in forward locomotion (AVBR) and backward locomotion (AVAL). Neural synchronization is shown in Fig. 23.5, obtained from (Kato et al., 2015).

Figure 23.5 shows that spontaneous network activity in the *C. elegans* brain is dominated by the synchronization of the neurons. Figure 23.5a shows the heat map of an exemplary whole brain Ca++ imaging dataset. Each row corresponds to a neuron, the color bar represents relative fluorescence change, and Fig. 23.5b shows the matrix of pairwise correlation coefficients averaged across four such worms. The correlation matrix in Fig. 23.5b is sorted by agglomerative clustering. Evident are two large synchronous neuronal ensembles of backward and forward neurons, respectively. The backward set of neurons is formed by interneurons AVAL, AVAR, AVEL, AVER, RIML, RIMR, AIBL, and AIBR, while the forward set of interneurons comprises AVBL, AVBR, RIBL, RIBR, among others. Each set of neurons supports the corresponding motor neurons: classes VA and DA for backward locomotion and motor neuron classes VB and DB for forward locomotion. Inside these large groups of neurons, smaller clusters are further synchronized, indicating partitions into fibers. These fibers require a more refined analysis of synchronization developed next.

Notably, brain activity under these conditions occurs spontaneously, in the sense that dynamics arise in the absence of any acutely delivered external and time-varying sensory stimulus. Clusters of synchronously active neurons, shown in Fig. 23.5b, generate activity patterns corresponding to behaviors such as forward crawl, reverse crawl, ventral turn, and dorsal turn. Importantly, fibers of synchronous network activity can be robustly observed across all of these conditions. Thus, synchronous neuronal ensembles are func-

tionally related. Since these dynamics and synchronies are not acutely triggered or bound by fluctuating external inputs, they must emerge as a property of the network itself.

Taken together, these features make this an ideal test case to scrutinize the symmetry hypothesis.

23.5 Synchrony and symmetry in the worm

Given that the structural connectome and neural synchronization in the whole body worm is readily available, we would expect that the forward arrow from structure \rightsquigarrow function can easily be validated in these animals, as was done for the TRN of *E. coli* in Chapter 20. However, this is not the case. Despite being fully mapped, no available connectome is complete, and the connectomes of different animals are different. For instance, if we analyze any available *C. elegans* connectome (Varshney et al., 2011) and calculate either automorphisms or fibrations, we will find just a few. Except in Fig. 23.3 where Morone and Makse (2019) have curated the data with a few hand-crafted modifications. However this is not feasibly in general. Yet there is an obvious synchronization of their neurons, as seen in Fig. 23.5. This means that this is an instance of the inverse problem of reconstruction, discussed in Chapter 22.

Ávila et al. (2025b) have developed a pipeline that utilizes the algorithm developed in Section 22.5 to repair the connectome of *C. elegans*, guided by the synchronization dynamics measured experimentally. Once this connectome is obtained, we can perform further perturbations to the neurons, such as ablation or inhibition, guided by fibration theory, to improve the understanding of the neural system.

The method repairs the connectivity structure of a baseline connectome, taken as the one curated by Varshney et al. (2011), and aims for a fibration symmetric connectome that reflects the observed synchronization. This method provides a means to idealize connectomes conceptually. The threshold for connectome modifications is set to a strict 25% variation based on inter-animal partial connectome differences (Hall and Russell, 1991), to ensure behavioral consistency despite neural network discrepancies. That is, the method modifies a baseline connectome to reproduce the experimentally obtained balanced coloring and it accepts the new connectome if the changes to the connectivity is below the 25% natural variation of the connectome. This conceptual framework proposes a novel inference scheme for reconstructing connectomes (or networks in general) while retaining accuracy in modeling dynamical neural behavior.

Figure 23.6 describes the pipeline of the method developed by Ávila et al. (2025b):

1. The process begins at Fig. 23.6a with a microfluidic device that maintains constant gas and pressure, creating non-disruptive conditions for proper fluorescent Ca++ imaging in *C. elegans* via a microscope and camera setup.
2. Time series data for activity traces of multiple neurons are recorded simultaneously, as shown in Fig. 23.6b. This procedure is performed on multiple worms under same conditions.

3. Various metrics that capture synchrony are applied to the recorded time series dynamics, producing correlation matrices that capture functional synchrony (Fig. 23.6c).
4. The correlation matrices shown in Fig. 23.6d are averaged across worms to obtain an average synchrony matrix.
5. A percolation process is then applied to obtain the functional network (Fig. 23.6e) using standard methods (Gallos et al., 2012b; Gili et al., 2025). Starting from a disconnected graph, links between nodes are progressively added in decreasing order of weight of the averaged correlation matrix.
6. Averaging the functional networks leads to a consensus matrix across different methods of synchrony (Fig. 23.6f).
7. Clustering algorithm is implemented to find a partition of synchrony clusters (Fig. 23.6g). Each neuron is assigned a color according to its cluster of synchrony. These colors are used as inputs to the repair algorithm in the next step.
8. We implement the mixed integer linear programming (MILP) algorithm of Section 22.5 to add/remove a minimal number of edges from the baseline incomplete connectome to produce an ideal fiber-symmetric network that reproduces the coloring in the consensus partition obtained in the previous step (Fig. 23.6h).
9. The algorithm produces a fiber-symmetric network with cluster synchronization that reproduces the experimental data. This network can be collapsed into a smaller representation, the base graph, where nodes belonging to the same fiber have isomorphic input trees (Fig. 23.6i).
10. For each consensus partition, a permutation p -value test is performed by permuting node labels and repairing the structural network 1,000 times. The frequency with which the initial partition outperforms permuted versions is measured by the modifications needed to create a fiber-symmetric network. The partition with the lowest p -value is chosen as the optimal solution (Fig. 23.6j).

23.5.1 Whole nervous system recording and motor neuron characterization

The setup of (Ávila et al., 2025b) records Ca++ activity of motoneurons involved in backward locomotion along the worm's length. Identification of each neuron is done by relying on a neuron dictionary such as NeuroPAL that is able to capture almost all nervous system activity with single-cell resolution (Yemini et al., 2021) (Fig. 23.7). This involves almost 300 neurons per recording, including the head ganglia, the complete ventral cord, and the tail ganglia (Fig. 23.7a). Figure 23.7c show a typical multi-neuron time series, with discernible calcium activity patterns and known neurons (Kato et al., 2015; Uzel et al., 2022) participating in the reverse motion. These include reversal interneurons such as AVAL/R and AVEL/R, which are major coordinators of the motor activity of dorsal and ventral reversal neurons, DA, and VA respectively.

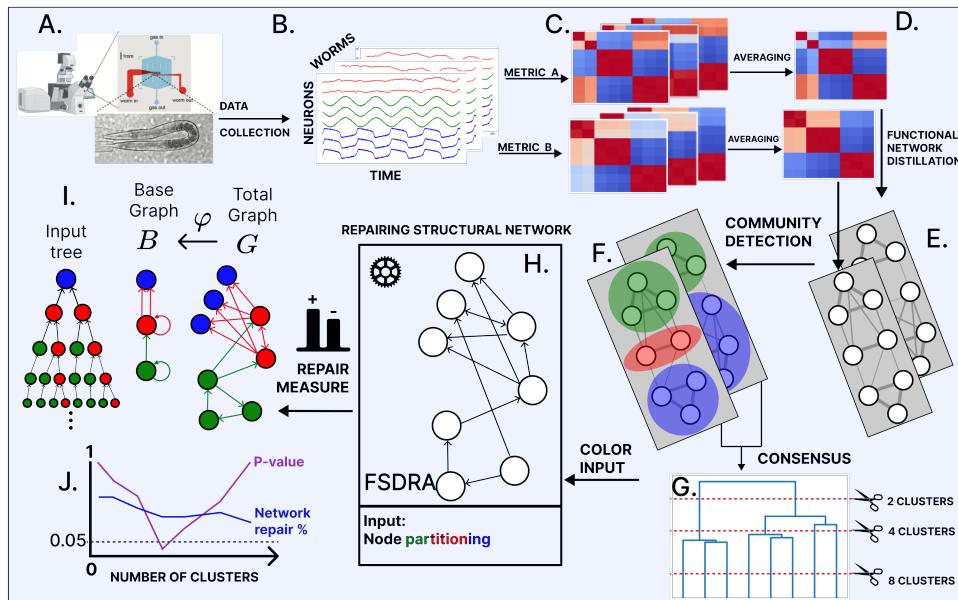


Fig. 23.6

Pipeline for the structural network repair of the backward locomotion of *C. elegans* based on neural recordings. This pipeline is quite general and can be applied to any dataset to infer the network from the synchronization. Figure reproduced from (Ávila et al., 2025b).

23.5.2 How to measure synchronization

The task of determining synchronization between two or more simultaneously recorded signals has been carried out in many different ways, depending on the characteristics that we are interested in measuring and determining their level of similarity. This results in a diverse array of techniques aiming to capture synchronization through different approaches (Schoffelen and Gross, 2009). Naturally, all this leads to a zoo of methods that try to capture the level of synchronization between signals through different aspects (Schoffelen and Gross, 2009). Many of the techniques have been discussed already in Section 20.3. Below, we expand on some methods more suitable for brain dynamics.

These techniques are generally classified into four primary groups based on their focus: time domain versus frequency domain, and methods that account for directional dependencies (like measuring causality in the signals) versus those that do not (like a correlation function). Some of these methods can be observed in Table 23.1.

Selecting the appropriate metrics can sometimes be difficult. It is better approach is

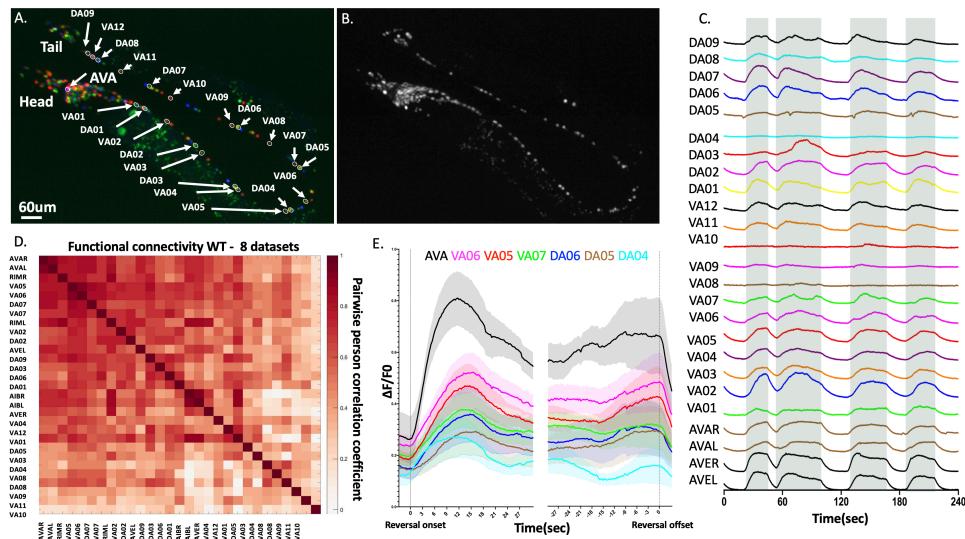


Fig. 23.7

Whole nervous system recording reveals synchronous motor neuron activity. (a) NeuroPAL labeled worm with selected neuronal cell class identities indicated. (b) Same worm as in (a) showing NLS-GCaMP6f labeling. (c) Activity time series (DF/F₀) of selected neurons; gray shading indicates reversal command states defined by AVAL activity. (d) Pairwise Level of Synchronization (*LoS*) among selected neurons. Each matrix entry is the average of $n = 3 - 8$ pairwise observations. (e), (f) Triggered average (\pm SEM) to reversal command onset (e) and offset (f), both defined by reference neuron AVAL. Shown are 3 example neurons of each VA and DA motoneuron class. Averages were calculated across $N = 3 - 8$ recordings including $n = x - y$ events. Figure reproduced from (Ávila et al., 2025b).

to use the information that these methods provide in a consensus manner to decipher the synchronization in the system.

23.5.3 Measures of synchrony

In the neuroscience community, correlation is the de facto synchrony measurement for neuronal data (Kato et al., 2015; Hallinen et al., 2021; Creamer et al., 2022). The Pearson correlation coefficient (20.3) is one of the more popular metrics to capture synchrony. Its main lure is that it captures whether two signals have the same distribution or whether they are monotonically related. Different correlation measures can be used, such as Spearman,

	Time Domain	Frequency Domain
With Directional Dependency	Cross-correlation Granger Causality Transfer Entropy	Phase Locking Value Directed Transfer Function Partial Directed Coherence
No Directional Dependency	Pearson Correlation Spearman's Rank Correlation Mutual Information	Coherence Spectral Correlation Phase Coherence

Table 23.1: **Synchronization measures in time and frequency domains.** There is a sea of measures used in the literature (Schoffelen and Gross, 2009). Each captures different aspects of synchronization.

Kendall, or Distance Correlation plus Covariance to aid in casting a wider net to capture linear or nonlinear correlation dependencies among signals.

Another popular measure of synchronization is the Phase-Locking Value (PLV) (Bruña et al., 2018; Lachaux et al., 1999), which measures synchrony in the phase rather than the amplitude of the signal. This metric is used in Chapter 25 in the human brain.

We also use the Level of Synchrony (*LoS*) introduced in (Phillips and Venkatasubramanian, 2011; Ávila et al., 2024) to determine how closely two signals are to perfect synchrony, as in (20.1). This is done for V_i , which measures the dynamic of a neuron using its membrane potential:

$$V_i(t) = V_j(t) \quad \forall i, j \in C_k. \quad (23.2)$$

Here, C_k is the synchrony cluster.

LoS evaluates how closely synchronized two signals are over time, by considering their instantaneous differences and scaling them through a parameter σ :

$$LoS_{ij} = \frac{1}{N+1} \sum_{t=0}^N \exp\left(-\frac{[V_i(tT/N) - V_j(tT/N)]^2}{2\sigma^2}\right). \quad (23.3)$$

Here, $N + 1$ is the total number of time steps in which *LoS* is applied between the signals from neurons i and j , and T is the total time interval considered. The range of the *LoS* function is $[0, 1]$, where a value of 1 indicates full synchrony, akin to (23.2) as exemplified by identically colored coded signals in Fig. 23.6b. A value of 0 indicates no synchronization. The parametric term σ serves as a scale to define a benchmark for closeness between two points in time.

This parameter deals with natural variations in biological systems. No two neurons are perfect copies of each other, so if two similar neurons at rest are stimulated by identical signals, their outputs (membrane potential) will naturally vary in intensity and phase by small amounts. With this concept in mind, we can implement multiple versions of the *LoS*, each with a different value for σ . When the value of σ is zero, each signal will be synchronous only with its exact copy, as the value of σ increases, it reaches a state in which all signals are synchronous with all others. Thus, an ideal value of σ lies somewhere between zero and an upper limit; the largest difference between signals at some time t should suffice.

23.5.4 Synchrony and correlation matrix

Once the particular metric of synchrony is chosen, either a Pearson correlation (20.3), a *LoS* (23.3), or any similar measure, an average correlation matrix is calculated. These matrices are first computed for each of the n subjects, capturing the synchronous activities observed within each individual as depicted in Fig. 23.6c. These individual matrices are then averaged in groups, as in Fig. 23.6d.

The simplest procedure to build the functional network follows a percolation process introduced by Gallos et al. (2012b), like that explained in Section 20.3.1. The functional entries are taken one at a time, starting with the largest value and continuing in descending order. The procedure halts when it incorporates the final neuron into the network, as represented by Fig. 23.6e, leading to a functional network as seen in Fig. 23.6e and exemplified in Fig. 20.3. More sophisticated methods of percolation can be used as well; see the analogous problem for gene coexpression data in Sections 20.3.1 and 20.3.2.

23.5.5 Clique synchronization

The functional network is used to find the synchrony clusters. Various methods, such as cluster and community detection algorithms, can be applied to extract these functionally synchronous clusters of neurons from a functional network, reviewed before. Two additional methods are briefly explained below.

The clique synchronization method introduced by Gili et al. (2025) decides that a node belongs to a clique if the average of its internal edges is bigger than any of its external edges. Any non-assigned node is forced into the clique from which it receives the highest average weight.

Ideally, perfect cluster synchronization is a non-overlapping, fully connected induced subgraph (clique) embedded in the functional network. Since ideal synchrony cannot be expected in real data we relax this condition, allowing the fully connected subgraph to be connected by weak inter-clique links. We define a *cluster synchronization N-clique* as an induced, fully connected subgraph of the functional network composed of N nodes that satisfies the following conditions:

$$\sum_{i < j}^{1,N} \sigma(x_i(t), x_j(t)) \geq \frac{N(N-1)}{2} \sigma(x_k(t), x_{k'}(t)), \quad (23.4)$$

$k = 1, \dots, N$ and $k' \in \mathcal{M}_k$

where \mathcal{M}_k is the set of nearest neighbors of node $k = 1, \dots, N$ not belonging to the clique concerned and $\sigma(x_i(t), x_j(t))$ is the chosen metric of synchrony between time series $x_i(t)$ and $x_j(t)$ of nodes i and j , respectively.

The clusters of synchronized nodes are again obtained by applying a percolation threshold procedure to the clique synchronization matrix. Starting from a disconnected graph, links between nodes are progressively added in decreasing order of weight in the correlation matrix (i.e., degree of synchronization), starting from the largest. A synchronized clique is found as soon as (23.4) is satisfied.

The largest clique is chosen when two partially overlapping cliques satisfy these conditions. If the size of these two cliques is the same, the clique with the largest edge weight average is chosen. The process stops when the weight of the links to be added does not allow further cliques to form. This process defines a hierarchy of cluster synchronization according to the order of clique appearance in the percolation process.

23.5.6 Modularity and community detection

Another method to find clusters of synchrony are Louvain community detection and any hierarchical clustering such as those discussed in Section 17.4. This kind of method decides the number of clusters and the number of nodes in each cluster by optimizing the modularity function (17.7), which is solely cluster-dependent.

The Louvain measurement at every execution assigns every neuron to its synchrony cluster. It proceeds by randomly grouping neurons into bigger clusters, accepting mergers only if the modularity value (17.7) increases, and halting once this value can no longer increase. Due to the stochastic nature of this process, each execution may lead to a slightly different result. For this reason, the Louvain method is executed 1,000 times for a distilled network that retains the partition with the highest measured modularity.

23.5.7 Consensus cluster synchronization

A consensus is created to leverage the information obtained from all the unique partitions across all methods, from Louvain to clique synchronization and metrics, from *LoS* to different correlations measures, and across parameters of these methods, following (Tian et al., 2022). All these matrices can be summed and normalized. The consensus matrix is shown in Fig. 23.8a.

From this consensus, we can obtain various partitions depending on where the dendrogram is sliced. In Fig. 23.8a the dendrogram is sliced at a value of 3 to obtain its three major groups observed in the consensus matrix. Each block is further divided into smaller diagonal blocks, indicating the solution with the least amount of modification to convert the collapsed Varshney connectome into a fiber-symmetric solution (5 clusters for a cutoff at 2).

23.6 Symmetry-driven reconstruction of the *C. elegans* locomotion connectome

The resulting clusters of neurons, found to be synchronous via the above procedures are used to repair the raw chemical synapse connectome of *C.elegans* for the backward locomotion gait shown in Fig. 23.8b. We use the linear integer program described in Section 22.5 to modify this binary network by adding and removing the fewest edges.

The goal is to obtain a network with a fiber partition satisfying the balanced coloring

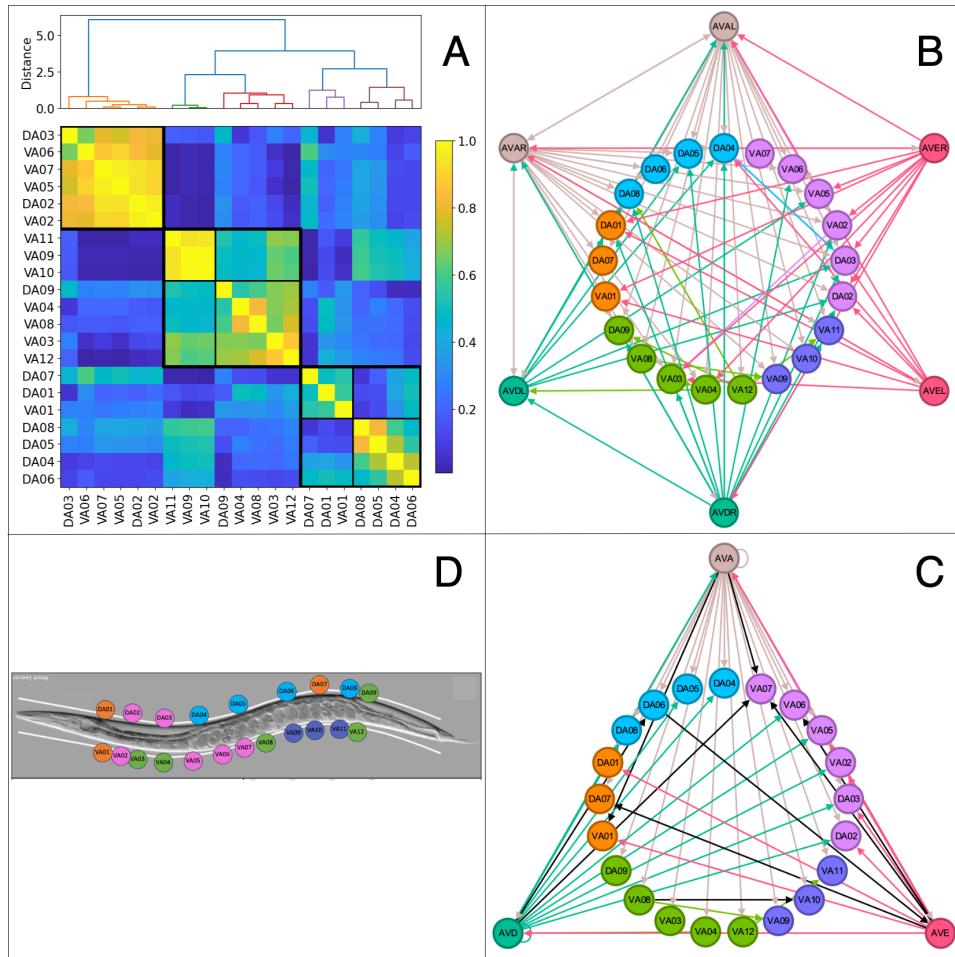


Fig. 23.8

Fibration reconstruction of the locomotion connectome. (a) The averaged consensus co-occurrence matrix shows (five) cluster synchronization of the neuron activity. (b) Original binary Varshney chemical synaptic network among backward motor-neurons and their three primary interneuron pairs AVAL/R, AVEL/R and AVDL/R. (c) Inferred locomotion connectome obtained by applying the inference algorithm to a collapsed version of (b). (d) Anterior-posterior distribution of motor neurons (DA_n, VA_n) involved in the backward locomotion in *C.elegans* colored by the clusters of synchrony. Figure reproduced from (Ávila et al., 2025b).

from cluster synchronization. Using the clusters found in the previous section, including consensus clustering, several repairs can be produced, one for each method used to calculate the clusters. The optimal solution for each type of clustering is chosen to be the one requiring the smallest number of modifications of the raw connectome by addition and removal of edges.

Figure 23.8c shows the final repaired network for the optimal reconstruction of the connectome that corresponds to consensus clustering. The final repairs are around 20% of the original edges, below the 25% threshold for acceptance of the solution. Figure 23.8d shows the neurons of the *C. elegans* connectome, with the balanced coloring obtained from the cluster analysis.

23.7 Perturbations and symmetry breaking by ablation and inhibition

Perturbing a network to determine its function is a central problem in biology and network science, encompassing brain/genetic network theory. The development of specific experiments to measure network dynamics by intervening with certain nodes is an experimental paradigm with vast consequences for understanding the functionality of the connectome.

Symmetries predict that symmetry-breaking by, for instance, ablation of a specific neuron or a group of neurons or any other perturbation such as optogenetics or transgenic inhibition, should lead to desynchronization of nodes in the affected fibers. These experiments are difficult to implement since it is desirable to manipulate more than one neuronal class to break left-right symmetry. Yan et al. (2017) used control theory to guide the manipulation of the connectome and obtained new functionalities. However, a network with strong symmetries is uncontrollable (Russo and Slotine, 2011). On the other hand, symmetry considerations can guide ablation experiments to break or create new synchrony patterns in the network.

Many of the observed network symmetries stem from the left-right symmetric body plan of nematodes. Permutations of left/right pairs (e.g., AVAL/AVAR) make up a large component of the symmetry operations. Thus, manipulating a neuronal class, e.g., AVA, will, in almost all cases, represent a symmetric interrogation. The vast majority of genetic drivers that are available for optogenetics or transgenic inhibition affects both the left and the right member of each class. In contrast, laser ablation is the only reliable method that allows loss of function of individual neurons, i.e., single members of a class.

Ablation experiments can be guided by theory so that they systematically search for the minimal ablations (ideally just one or a few neurons) that have a maximal effect on breaking symmetry in the network. Fibrations and symmetry groups predict that breaking symmetries by laser ablation should lead to specific asynchronies in neuronal activity patterns, while control ablations that do not affect symmetries should not affect synchronies but could, perhaps, affect other activity patterns.

In the case of control ablations, there can still be other effects on network activity, such as altered signal amplitudes and frequencies, and the theory predicts specific effects on synchronies. An example of this kind is given in (Kato et al., 2015). When both AVAL and AVAR are inhibited, animals are largely unable to execute reverse crawling actions due to a disconnect of the brain from the motor sector. However, network synchronies across AVA connections remain largely intact.

The theoretical reconstruction of the connectome suggests performing systematic *in sil-*

ico searches, providing examples to apply laser ablation procedures, and then to perform whole nervous system activity recordings in worms, which could lead to a complete characterization of synchronies in the connectome. The results of these experiments would provide a rigorous test of the symmetry framework.

23.7.1 Future work

The use of integer linear programming to adjust connectomics data based on functional imaging, demonstrate the use of a novel computational approach to neuroscientific research. This technique refines existing neural network models and provides a quantifiable method for aligning theoretical predictions with empirical observations. As shown in Fig. 23.8, this points towards an anatomical organization of the chemical network that sustains the smooth sinusoidal movement of the worm while moving backward. Our results suggest that there could be at least three major groups of activity (anterior, midbody, and posterior) that relate to the shape of this sinusoidal movement. Ultimately, our results also suggest that this activity can be further subdivided into five groups, which could be related to how the different motor neurons connect to different muscle segments. Future ongoing experiments imaging muscle activity in immobilized and freely moving conditions will surely help to clarify this hypothesis.

The results of Ávila et al. (2025b) have substantiated the pivotal role of connectome structure, specifically fibration symmetries, in orchestrating neuronal synchronization in *C. elegans*. The refined understanding that structural connectomics provides to underpin significant aspects of neural functionality, furthers our grasp of the physical basis of neural synchronization. This synchronization is crucial since it forms the foundation for coordinated motor outputs and behavioral responses in organisms. The use of advanced calcium imaging and graph theory to correlate these structural motifs with dynamic patterns of neural activity offer a compelling model for predicting neuronal behavior based on underlying anatomical data.

Thus, the implications are not restricted to *C. elegans*. The principles of neuronal synchronization, facilitated by connectomic structures have analogs across bilaterian species, including mice and humans (Gili et al., 2025) as investigated in the next two chapters. Investigating these principles in more complex nervous systems reveals insights into how brain-wide synchronization patterns contribute to complex behaviors and cognitive functions.

Fibration Theory of the Brain II: the Minimal Engram in Mice

In 1949, Donald Hebb postulated that the substrate of memory resides in the strengthening of synaptic connections, which occurs when the neurons concerned are active at very similar times. Hebb's principle states that if one neuron firing consistently causes another to fire, then the strength of the connection increases. In this causal form, Hebb's postulate has been largely confirmed experimentally. Important though the postulate has been, it left unanswered the fundamental question of how the memory engram, as a whole, is engraved in the structure of the network connectome when new information is encoded. What are the fundamental building blocks in the brain network that form memory engrams to assimilate new information into our knowledge base? This chapter shows how engram formation is reflected in the fibration symmetries of synaptic connections.

24.1 One-to-many relation between structure and function in the engram

It has been proposed by Park and Friston (2013) that the brain's functional activity selectively engages a subset of connections within the connectome 'highway' to operate in various functional states. This 'one-to-many' degeneracy in the structure-function relationship allows for the emergence of diverse functional states (e.g., resting, memory, movement) from a single, consistent connectome architecture. Although the postulate has been important, it left unanswered what the fundamental theoretical framework that links the connectome's structure to its function is. In this context, we focus on the formation of 'memory engram networks,' which serve as the neural substrate for storing and retrieving information related to experiences. We first show that mesoscopic brain areas exhibit covariance in the expression of immediate-early genes (*c-Fos*) across different animals, which is, as discussed in Section 20.3, a form of cluster synchronization, not in time, but across animals.

We utilize this synchronization to infer the substrate engram network by applying the inference algorithm developed in Section 22.5. The theory yields falsifiable predictions that can be tested experimentally, thereby assessing the applicability of the entire symmetry framework in describing the brain.

The fibration reconstruction predicts that the Agranular insula, both its dorsal and ventral aspects, along with the basolateral amygdala, play a crucial role in the particular engram network. We experimentally validate this prediction by using targeted pharmacogenetic

inactivation of these areas, demonstrating significant impairment in memory formation, retrieval, and performance on behavioral tasks compared to control experiments.

Thus, the symmetry framework may serve as the missing link to connect function and structure and could also be a valuable tool for assessing the role of essential nodes in the broader context of synchronous systems, extending from other brain connectomes to genetic networks in health and disease.

24.2 Where in the connectome are memories stored?

In 1921, evolutionary biologist Semon (1921) proposed the idea of an *engram* as '*the enduring though primarily latent modification in the irritable substance produced by stimulus*'. His work inspired generations of neuroscientists to join the search for a memory fingerprint in the brain.

A large body of experimental evidence supports the idea that memories are made, or encoded, through the strengthening of synapses in distributed associative circuits (Rolls and Treves, 1998; Roy et al., 2022; Poo et al., 2016). These circuits consist of interconnected excitatory and inhibitory neurons in different brain sub-regions or network nodes. However, memory is not a single unitary process, and it is neither located in one particular place in the brain nor static in time. Engrams are known to be distributed across different brain regions, defining separate neuronal assemblies (which represent multimodal aspects of a given memory). How these distant assemblies are coordinated to form a specific engram is still unknown.

Memory is, rather, a distributed and dynamic phenomenon that enriches our knowledge base with new experiences. As we explore an unknown place, for instance, a new memory of the context is formed in the brain. The spatial landmarks, the pathways, the presence of potentially useful resources (e.g., food), and the timing of the exploration episode are stored in memory. After some time, navigating that environment to collect food from the known resources and even planning the time of the excursion and the pathway ahead are aided by recollecting the stored memory traces. These later excursions do not require significant encoding of new information until, for instance, one food resource is extinguished. At that moment, memory needs to be updated to adapt to the new environmental configuration.

How do these memory engrams evolve across such contingencies? What is the network structure of the memory engram at the moment of encoding, during retrieval or updating? More importantly, which network building blocks critically define information routing in the memory engram, how is the engram engraved in the structure of synaptic connections, and how do engrams evolve during different learning cycles? The integrated, distributed, and specialized network activity leads to synchronization in the information processing system, and, like any synchronization phenomenon, it is expected to be supported by the symmetries of the underlying engram connectome.

24.3 How to measure engram synchronization

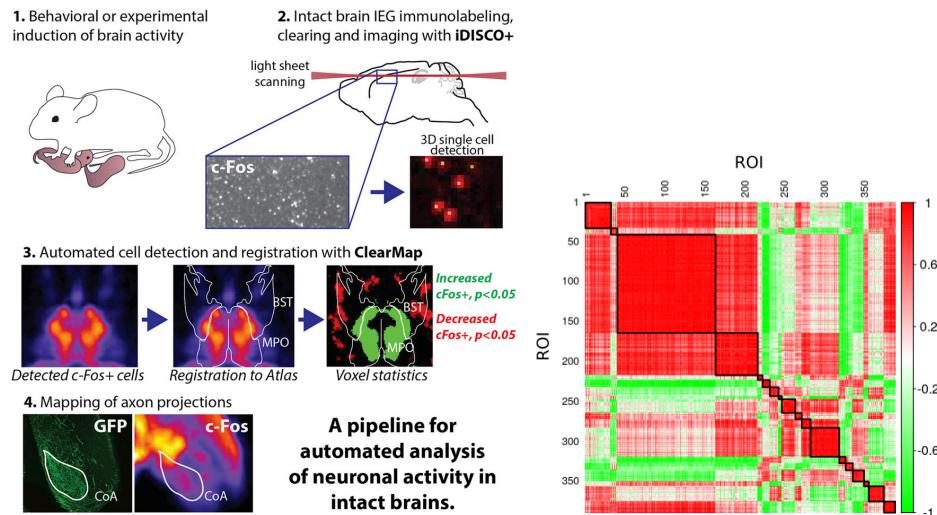
While memory is thought to be encoded through modifications of the weights of synaptic connections (Hebb, 1949), it is not known how the multiple and dispersed cell assemblies activated by a particular experience are engraved in the structure of the connectome. Time is important for establishing associations at the cognitive level, as is the timing of inputs for stabilizing synaptic modifications. Thus, synchronization between experience-relevant neuronal populations is fundamental for building memory engrams.

If the formation of the engram is a synchronization phenomenon among neural populations supported by the modification of the weights of the synaptic connections, it is natural to enquire whether the underlying symmetries of the connectivity of the neurons in the engram could determine the building blocks of the memory networks. This leads to the synchronization of experience-relevant neural populations and their binding into coherent engrams, thereby supporting new memory formation. Finding the building blocks of engram synchronization reveals the distributed nature of memory. It could also reveal the mechanisms for segregation of synchronized function in the presence of integration of information in the entire brain, which is a fundamental problem in neuroscience (Tononi et al., 1994; Gallos et al., 2012b).

Synchrony in the engram can be tested in a real memory engram network recorded from awake and freely behaving rodents involved in learning tasks. Animals acquire ‘knowledge’ no less than humans, so it should be possible to use animal models of memory formation to secure a mechanistic understanding through causal experiments.

Modern memory engram technologies, aided by molecular biology, make it possible to tag activated neurons in the engram network in time windows that can be experimentally controlled. It has been shown that activation of neurons is accompanied by the expression of plasticity related immediate-early genes (IEGs) such as c-Fos (Vann et al., 2000; VanElzakker et al., 2008; Josselyn et al., 2015). In this way, for instance, the engram cells associated with the formation of a contextual fear memory can be genetically tagged (Lei et al., 2007; Dheeraj et al., 2016) and revealed (with fluorophores), and the activity can be manipulated (with c-Fos-driven expression of optogenetic tools). Later reactivation of c-Fos positive neurons tagged during a particular experience evoke the memory encoded during that experience, demonstrating that c-Fos labeled neuronal assemblies constitute a memory engram (Ryan et al., 2015).

Using this technology, Renier et al. (2016) established an experimental pipeline to measure engram formation in mice, shown on the left of Fig. 24.1. The right of Fig. 24.1 shows the existence of synchronized neurons in the engram using these data (Renier et al., 2016). In each of these experiments, a mouse was placed in a cage to explore a new environment ($n=4$ mice). Then, mice were sacrificed and processed histologically to find the specific set of neurons that were active during the task. This was done by staining the c-Fos immediate-early gene, which correlates with neuronal activity. High-resolution optical imaging with light-sheet microscopy and immediate-early genes (IEGs) expression (c-Fos) allows the quantitative assessment of neuronal assemblies in the complete brain, formed in response

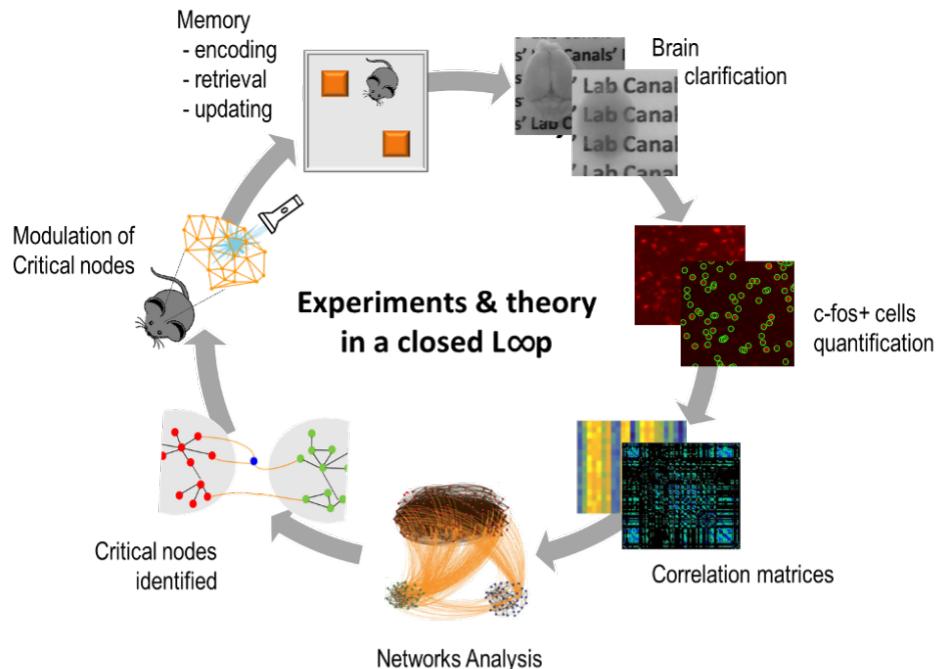
**Fig. 24.1**

Synchronization in engram formation in mice. *Left:* Pipeline designed by Renier et al. (2016) for the automated analysis of neuronal activity in intact brains. *Right:* Using c-Fos expression data obtained from (Renier et al., 2016), we show the synchronization of ROIs in the brain engram network made of 380 ROIs in mice. Similar synchrony is observed at the global level in the memory network under LTP in (Del Ferraro et al., 2018). The structural origin of the synchronization in the memory engram can be thought of in terms of symmetries of the underlying connectome. Figure reproduced from (Renier et al., 2016). Copyright © 2016, Elsevier Inc.

to a particular behavior. Following brain clarification (making tissue transparent), the active c-Fos⁺ cells were imaged and quantified using available tools (e.g., ClearMap).

Next, the number of cells in different regions of interest (ROIs) was computed, based on the Allen Brain Atlas—an open-source parcellation atlas for the mouse brain (Oh *et al.*, 2014). Using the sample data, a set of 380 ROIs was considered to cover the complete brain to compute a matrix of Pearson correlations across all ROIs. Figure 24.1 (right) shows the resulting correlation matrix with hierarchical clustering across the 380 ROIs. The correlation matrix, analyzed by typical clustering methods, shows evidence for a set of highly correlated synchronized ROIs supporting distributed synchronization at the mesoscale.

Similarly, Del Ferraro et al. (2018) observed high correlations in long-term potentiation (LTP) experiments in rodents. These synchronies are manifestations of symmetries at the meso- and macro-scale analogous to those observed in the synchronized locomotion of *C. elegans* discussed in Chapter 23 at the neural level. Del Ferraro et al. (2018) confirmed synchronization in distributed memory networks using fMRI data and pharmacogenetic interventions in anesthetized rodents based on the synaptic plasticity paradigm of LTP as a laboratory model of memory formation in rats.

**Fig. 24.2**

Closed-loop design to study memory engram synchronization and symmetry (García-Hernández et al., 2025). Genetically and histologically labeled memory engram networks are reconstructed from clarified brains, critical nodes are identified, and their activity is modulated with optogenetic tools. Closing the loop, the outputs of these manipulations are read out at the cognitive and network levels.

García-Hernández et al. (2025) study how engram synchronization is manifested in the symmetries of the mesoscopic connectome in mice. They show that the binding of dispersed cell assemblies into memory engrams is controlled by a set of building blocks for network synchronization that can be identified by fibration theory. An extrinsic value of this analysis is the application of the mathematical tools for neuronal network synchronization and integration to understand different cognitive functions and pathological alterations.

García-Hernández et al. (2025) recorded memory engram networks from awake and freely behaving rodents involved in learning tasks of (1) memory encoding (engram formation), (2) memory retrieval (engram reactivation), and (3) memory updating (engram remodeling). Following a closed-loop design between theory and experiment (Fig. 24.2), they used models of memory formation to secure a mechanistic understanding with causal experiments. They implemented the brain clarification technique together with dynamic c-Fos quantification (Fig. 24.3). They performed $n = 7$ mice experiments in which engram cells were recorded at two-time points, one during memory encoding (genetically labeling c-Fos expression) and the other during memory updating and recall (staining c-Fos with immunolabeling). This method, developed in (De Nardo et al., 2019), provides dynamic

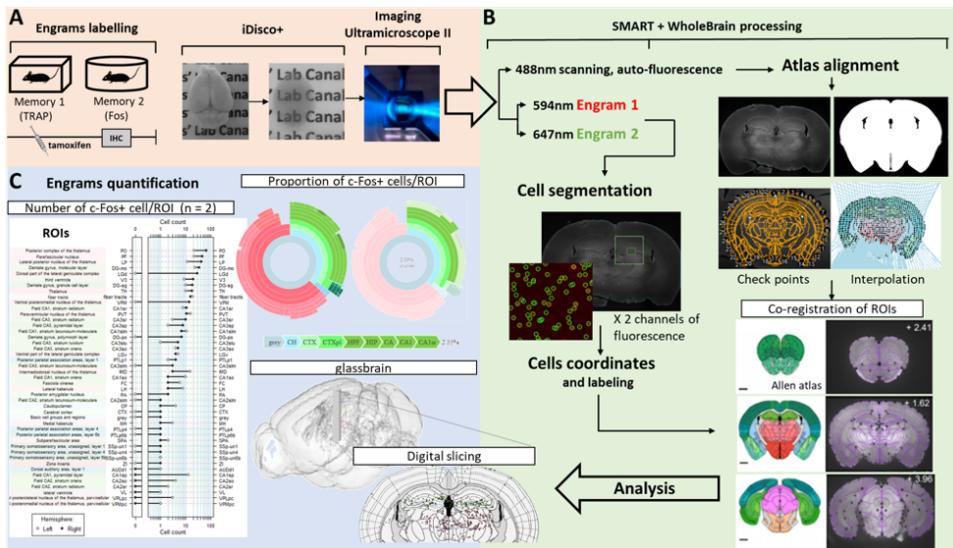


Fig. 24.3

Whole-brain c-Fos+ cell quantification and memory engram network

construction (García-Hernández et al., 2025). **(a)** After labeling engram cells at two time points of memory formation, brains undergo iDisco+ procedures for tissue clarification and light sheet microscopy. **(b)** Imaging is done at three different wavelengths. At 488 nm, the autofluorescence of the tissue provides a complete 3D anatomical image optimal for atlas alignment (Allen Brain Atlas). At 594 nm and 647 nm, we image the c-Fos+ cells (genetically or immunohistologically labeled) of the two memory engrams, respectively. **(c)** The number of positive cells in each region of interest (ROI) defined in the atlas is counted per animal ($n = 7$). The size of the ROIs used to build the network is dynamically adjusted (circular hierarchical plot) to define the spatial resolution and to test the impact of coarse-graining in the final results. ROIs differentiate left and right hemispheres (dot plot). Finally, this information is used to build correlation matrices and perform network analysis to obtain the clusters of ROIs synchronized by the engram.

information on engrams since each animal provides two temporal instances of the engram map. The first is at the inception of the memory; the second is when it is recalled and updated, and the original information is enriched.

High-resolution optical imaging with light-sheet microscopy and IEGs expression allows the quantitative measurement of activated neuronal assemblies in the complete brain of the mouse in response to a particular behavior (Renier et al., 2016; De Nardo et al., 2019; Roy et al., 2022). c-Fos measurements allow for studies in freely behaving animals and provide high sensitivity to neuronal activity changes. After performing the behavioral experiment, animals are sacrificed, and the perfused brains are extracted and processed for tissue clarification. Then, genetically labeled or immunohistochemically stained c-Fos+ nuclei are imaged under a light-sheet microscope. The 3D brain images of each subject

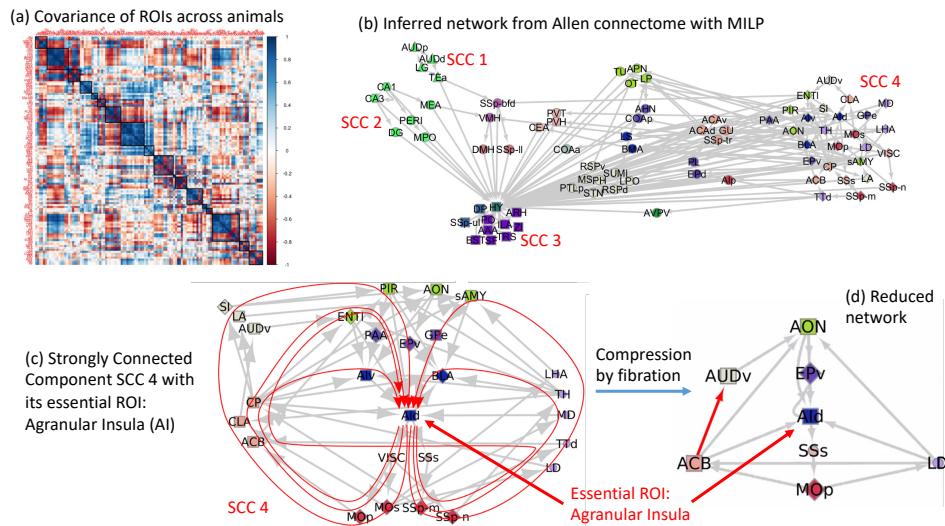


Fig. 24.4

Inferring the engram connectome. (a) Covariance matrix between activated ROIs showing the hierarchical clustering into 18 clusters of coherence across animals. (b) The memory engram: Using the clusters from (a), MILP then infers the substrate connectome to which graph tools are applied to find essential nodes. Four SCCs were identified; SCC 4 was the most unexpected. (c) SCC4 is analyzed, and the Agranular Insula is identified as the central node of this SCC and a primary candidate for disrupting its integrity. (d) By applying a fibration, the network can be compressed by collapsing the colored clusters into one representative node of the cluster for further graph processing. Results obtained by (García-Hernández et al., 2025).

are co-registered to a common anatomical template. This step is critical since it makes it possible to identify and label the same ROIs consistently in the population of subjects used to construct the network.

24.4 Inferring the engram network

García-Hernández et al. (2025) compute a covariance matrix across 380 ROIs as defined in (Renier et al., 2016) by averaging over all animals considered. The resulting covariance matrix is shown in Fig. 24.4a, which provides evidence of coherence in the engram formation across animals. The analysis is very similar to the gene coexpression correlation analysis in Chapter 20 and the synchronization analysis in Chapter 23 in *C. elegans*. However, the small number of animals used in this study ($n = 7$) requires massive regularization techniques to make the correlations to be significant. It must be said that $n = 7$ is the largest c-Fos study so far.

The bundles of axonal tracts between ROIs are obtained from the Allen Brain Atlas connectome (Oh *et al.*, 2014). These tracts constitute the baseline connectome of the whole mouse brain. It represents the available highway system composing the underlying information routes of the brain involved not only in memory but in any functional task. However, which routes of this highway are utilized depends on the type of task for which the brain is responding. The main hypothesis is that the brain's functional activity utilizes a subset of the links available in the highway connectome to operate in each functional state.

The ‘one-to-many’ degenerate structure-function postulate of Park and Friston (2013) allows the emergence of diverse functional states from a unique static connectome architecture. In the present case, it means that, given the Allen baseline connectome (one), different subsets of this connectome mediate different engrams (many).

To uncover this engram network, we use the inference algorithm developed in Section 22.5 by assigning a color to each cluster of synchrony obtained from the covariance matrix. The baseline connectome is considered fairly complete, although some missing links might exist. Therefore, the algorithm allows for the removal of links as well as addition (with a lower probability).

24.5 The minimal engram network

Figure 24.4b shows the resulting inferred engram network. Several steps of reduction have been applied here. First, we identify 72 ROIs (out of 380) significantly activated in the engram. Then, we group these ROIs in 18 clusters obtained through the clustering of the covariance matrix in Fig. 24.4a. Each cluster is assigned a color. We then reconstruct the baseline Allen connectome Fig. 24.4b to infer a network that satisfies the balanced coloring based on the 18 clusters. After obtaining this connectome, graph analysis follows.

The first step is to remove (trivial) ROIs that do not contribute to the integrity of the network. These are nodes that have no out-degree. They only receive information but do not transmit it. The next step is to obtain the graph's SCCs, where information travels across cycles. We find that there are 4 SCCs in the network (Fig. 24.4b). Two of them are expected: one is dominated by the hypothalamus (HY in SSC 3), which is a super hub receiving signals from all SCCs, and another by hippocampal CA1 and CA3, which is central for spatial memory formation. All of the SCCs send information to SCC 3, both directly and indirectly, through paths that cross other ‘bridge’ nodes. In fact, SCC 3 acts as an information sink, receiving input from many ROIs. The rest of the ROIs are not in SCC, but they are feed-forward connectors of the SCCs.

We further do a network analysis of SCC 4 using different centralities to find the essential nodes of this component (Makse and Zava, 2025). The essential nodes that hold this SCC together are part of the insular cortex, more specifically, the Agranular Insula (AI) on its dorsal (AId) and ventral (AIv) aspects, which, together with the basolateral amygdala (BLA) form a fiber at the center of SCC 4. This is also a hub of SCC 4 and has high betweenness centrality since there are many paths, the two red cycles in Fig. 24.4c, that go through this fiber. Upon inactivation of the AI, the whole SCC is disintegrated, revealing the essential

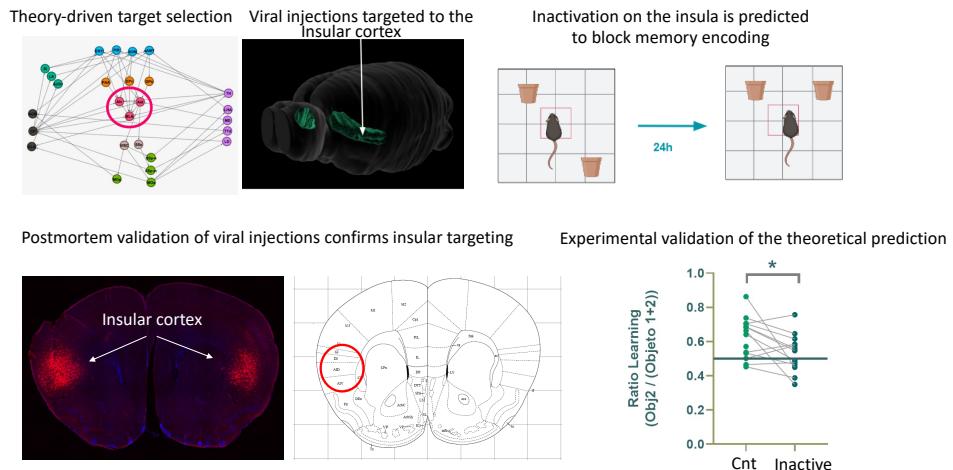


Fig. 24.5

Disrupting the minimal engram. Inactivation of the insula using 13 mice. Results obtained by (García-Hernández et al., 2025). After determining the target coordinates based on mouse anatomy, DREADD-expressing viruses were injected to inhibit the insula. Postmortem verification (red staining in the histology and red circle in the anatomical atlas) confirmed the accuracy of the procedure. In these animals, memory formation occurs normally when they encode contextual information with an intact insula (Control, light green dots). However, when the insula is inactivated ('inactive'), the animals lose the ability to encode new memories (dark green dots), as shown in the bottom-right figure. The learning ratio on the right quantifies the animal's exploration of the two objects in the arena, with higher ratios demonstrating a better memory representation of the two objects' location. The asterisk denotes statistical significance ($p > 0.05$ paired t-test).

character of this fiber. The network can be reduced for further processing by collapsing each of the fibers into the base, as seen in Fig. 24.4d.

Using the knowledge amassed in the entire book, we have obtained the '*minimal engram*' network. In this network, each supernode is a SCC. SCC 1, 2, and 4 send information to the sink hypothalamus-SCC 3. Connector nodes control and act as intermediates between these SCCs. Inside the SCCs, a cycling structure is found composed of memory toggle-switch building blocks. Amazingly, the structure of this minimal engram is quite similar to the minimal TRN uncovered in Fig. 19.1.

The analysis predicts that upon inactivating the AI, the SCC 4 would disintegrate, and memory formation would be dramatically impaired, in comparison to the inactivation of other nodes in the same SCC or others nodes, which are defined as non-essential. Inactivation of non-essential nodes may not produce a large effect since the network contains other pathways to transmit the information (for instance, other cycles in SCC 4).

García-Hernández et al. (2025) confirmed experimentally this prediction by pharmacological

logical inactivation ($n = 13$) of the insular cortex neurons identified by the graph model. To achieve inhibition of this region, they use parvalbumin (PV)-cre transgenic mouse, enabling cell-type-specific protein expression via adeno-associated viruses stereotactically injected into the target area, as done in the previous studies (Del Ferraro et al., 2018). Parvalbumin-expressing neurons are inhibitory interneurons, so activating this population induces strong inhibition in the insula. The results, presented in Fig. 24.5, demonstrate that insula inhibition negatively impacts memory formation. Figure 24.5 bottom right shows that when the insula is inactivated ('Inactive'), the animals lose the ability to encode new memories (dark green dots) in comparison with control experiments ('Cnt').

This result offers promising evidence for the symmetry theory inference of the network and the identification of critical nodes within it. It represents an encouraging first step in demonstrating that the symmetry approach can yield falsifiable predictions that can be tested experimentally. Numerous future analyses are expected. For example, it would be valuable to clarify whether this type of manipulation causes a disconnection of a significant component, such as the SCC 4 of the memory engram.

Overall, fibration appears to be an effective theory for bridging the gap between structure and function, helping to identify essential nodes in the network.

Fibration Theory of the Brain III: the Human Language Network

Understanding the human brain organization is the final frontier for the applicability of fibration theory. This chapter explores the relationship between structural connectivity (connectome) and the emergent synchronization of mesoscopic regions of interest in the human brain network. Among the allowed patterns of structural connectivity, synchronization elicits different fibration symmetry subsets according to the functional engagement of the brain. The resting state of the brain is a particular realization of the cerebral synchronization pattern characterized by a fibration symmetry that is broken in the transition from rest to a task engaged in language.

25.1 Structure and function in the human brain

The structure \rightsquigarrow function relation has been deeply studied in the human brain at the mesoscopic level (Park and Friston, 2013; Friston, 2011). Studies use diffusion and functional MRI to correlate white matter tracts to the functional coupling between the appropriate region of interests (ROIs) (Honey et al., 2009; Koch et al., 2002; Greicius et al., 2009; van den Heuvel et al., 2009; Hagmann et al., 2008). Previous studies have looked at correlations between structural connectivity (SC, e.g., from diffusion tensor imaging, DTI) and resting-state functional connectivity (rsFC, from fMRI) for anatomically defined ROIs (Honey et al., 2009; Koch et al., 2002; Greicius et al., 2009; van den Heuvel et al., 2009; Hagmann et al., 2008). While the SC resembles the rsFC, two ROIs can be structurally connected but not functionally related, and vice versa. Thus, these fMRI studies have demonstrated that anatomical connectivity does not always translate into functional connectivity.

Deco et al. (2013) studied the structure-function relation via theory and modeling. A theoretical framework to study the link between anatomical structure and resting-state dynamics were proposed based on DTI tractography and parcellation to provide the neuroanatomical structural network averaged across subjects. A neurodynamical model is used to model BOLD signals in resting and task-based cognitive states. The model is validated by comparing predicted spatiotemporal patterns with empirical functional connectivity data.

When the graph has no structure—think, for instance, of an Erdős-Rényi graph wherein nodes are connected at random—nothing can be said about its function. Away from randomness, many structures have been found in graphs. Neural graphs display modules, motifs, hubs, small worlds, fractality, and others (Bullmore and Sporns, 2009; Rubinov and Sporns, 2010). However, most of these structures say little about the function of the network.

However, when symmetries determine the structure of the graph, the relationship between

structure and synchronization becomes clear. If we identify synchronization with function, then a structure-function relation can be studied from a theoretical point of view. In this chapter, we explore this hypothesis at the mesoscopic scales measured by fMRI activity in the human brain. We show how symmetries of the mesoscale connectome (white matter tracts) can explain the synchronization of brain activity at scales measured by fMRI in the human brain.

This chapter develops the symmetry viewpoint of the human brain, letting empirical activity synchronization drive reconstructions of the connectome based on symmetry fibra-

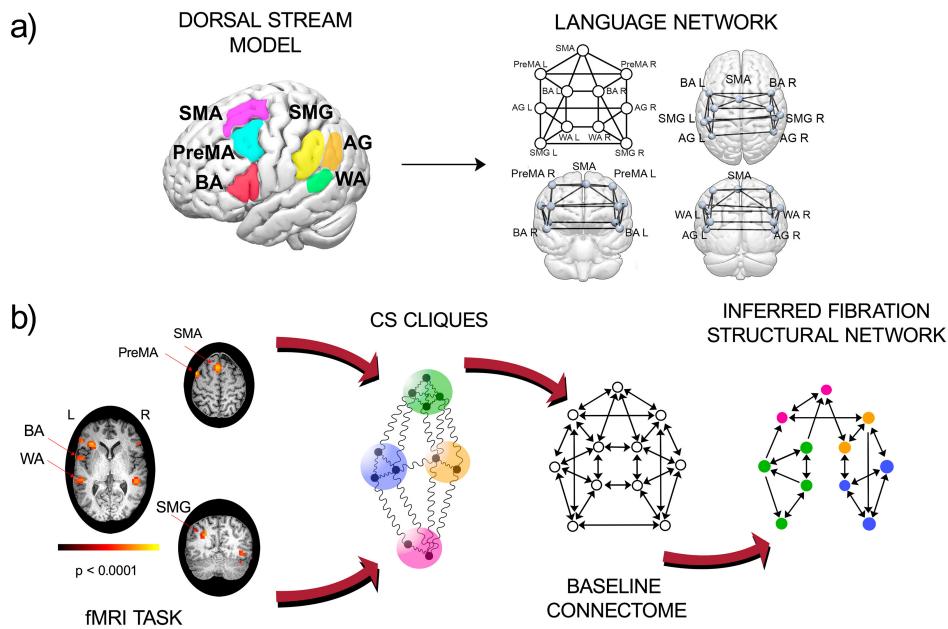


Fig. 25.1

Synchronization-guided inference algorithm. We exemplify the algorithm in the case of a brain network. (a) Left: Nodes of the primary language network are given by the dorsal stream of the dual-stream model localized in the 3D brain. Right: dual (dorsal) stream baseline connectome showing the fiber tracts between the ROIs in (a). (b) Pipeline for inference of the structural network from dynamical data. Left: fMRI images for a task or resting state over many subjects are taken as inputs to calculate the group-average cluster synchronization cliques among the nodes. The clusters are identified with the balanced colors or fibers in the baseline connectome. A mixed integer programming algorithm is employed to optimally infer the structural network (right) that sustains the coloring cluster pattern obtained from the dynamics. Figure reproduced from (Gili et al., 2025).

tions, and inferring how communicability shapes communication patterns among regions of the human brain using the reconstruction algorithms of Chapter 22.

Our contention is not that symmetries should describe human connectomes exactly. Instead, these symmetries must be realized in biology in an approximate way as quasipermutations or pseudobalanced colorings, as described in Section 13.5. Synchronization in the fMRI activity drives the inference of an ideal connectome that has vestiges of an underlying symmetry common to all subjects, shedding light on the engagement of the brain in different functional states.

25.2 Human language network

Activity in the human brain occurs at many scales. Spatial scales range from nanoscales at synaptic connections to systems of neurons and their organizations, to large-scale network structures such as cortical and sensory-motor networks. Temporal activity also occurs on widespread timescales, ranging from the milliseconds of action potential transmission to the minutes to a lifetime of large-scale plasticity changes triggered by experience and memory.

There are many experimental ways to interrogate brain activity at these different scales. Functional magnetic resonance imaging (fMRI) lumps all these scales together into a blood-oxygen-level-dependent (BOLD) signal. The BOLD signal is not a direct measure of neuronal activity, as given, for instance, by action potentials. It is an indirect measure of neural activity that measures changes in deoxyhemoglobin induced by alterations in blood flow and levels of oxygen in the brain. These changes are coupled to neuronal activity by neurovascular coupling.

BOLD signals can be used to map the functional connectivity of ROIs in the brain at scales of a few millimeters and a few seconds. In the human brain, activity is measured in an fMRI voxel, which is typically around one mm^3 and contains about a few hundred thousand neurons per voxel in the cerebral cortex. Whole brain fMRI imaging interrogates the brain at these intermediate temporal and spatial resolutions, but it is not sensitive to the msec temporal scales of neural activity.

To understand these data, we can examine the correlation functions of BOLD activity, as discussed in Chapter 20. This approach provides evidence of synchronization in the brain at broader scales. In this chapter, we investigate the phenomenon of synchronization in the human brain during a language task and demonstrate its connection to the underlying symmetries in the structural connections of white matter tracts.

Gili et al. (2025) have analyzed ($n = 20$) fMRI networks and DTI from healthy individuals and brain tumor patients at Memorial Sloan Kettering Cancer Center (MSKCC) in New York City. These subjects performed a language task and resting state (RS), allowing the construction of networks associated with expressive language. Language function has been widely studied using both resting state and task-based fMRI. In task-based fMRI, we can be reasonably sure of the functional activation in the brain concerned (Holodny et al., 2002; Li et al., 2019, 2020, 2021). The pipeline of the analysis is shown in Fig. 25.1.

The subjects perform two different language-associated tasks, as well as the typical

resting state protocol. The task-based protocol emphasizes a language function known to be associated with the left frontal lobe. Phonemic fluency (letter) and verb generation tasks are applied as part of the presurgical language task panel for preoperative planning for tumor surgery at MSKCC. During the phonemic fluency task, subjects generate words that begin with a presented letter (for example: a patient presented with the letter ‘A’ may generate words such as ‘apple’, ‘apron’, or ‘ashtray’). Subjects are also instructed to lie in the scanner with their eyes open, to try to think of nothing in particular, and to keep fixating on a central cross on a screen during RS.

According to current dual-stream models of language, different ROIs are involved in the language task (Hickok and Poeppel, 2007; Hickok, 2022). The areas that are most responsive in language task paradigms have been studied in (Li et al., 2019, 2020, 2021). In tasks that are focused on language production, only regions of the dorsal stream are part of the analysis. Specifically, the most current dual-stream model of Hickok (2022) asserts that the following anatomically defined ROIs of the dorsal stream are involved in these tasks: supplementary motor area (SMA), premotor area (premotor, left and right), supramarginal gyrus (supramarginal, left and right), Broca’s Area (BA, left and right), angular Gyrus (angular, left and right), Wernicke’s Area (WA, left and right).

The BA and WA are the primary areas responsible for language expression and comprehension. The SMA has largely been considered to be involved in controlling speech-motor functions. It has also been shown by Hertrich et al. (2016) that the SMA performs several higher-level control tasks during a speech communication and language comprehension. The angular gyrus is associated with complex language functions (i.e., reading, writing, and interpreting what is written), and the supramarginal gyrus is involved in the phonological processing of highly cognitive tasks. Processing an action verb depends in part on activity in a motor region that contributes to planning and executing the action named by the verb, and the premotor cortex is known to be functionally involved in the understanding of action language (Chang et al., 2023).

25.3 Synchrony in the human brain at the mesoscale scale

Cluster synchronization can be determined using the procedures discussed in previous chapters. In addition, Gili et al. (2025) has developed a method specifically tailored to fMRI data, as follows.

First, we use standard methods to build the functional network from the time-dependent, fMRI-measured BOLD signal. For a single subject, we measure synchronization via the Phase Locking Value (PLV) (Bruña et al., 2018; Lachaux et al., 1999) for the BOLD time series from each ROI pair. This metric captures phase synchronization, ignoring the amplitude. Thus, two signals with different amplitudes can still be synchronized by their phases. BOLD time series were extracted from all voxels in a sphere of radius 6 mm centered on the coordinates of each ROI. Each ROI was composed of 123 voxels. Once time series

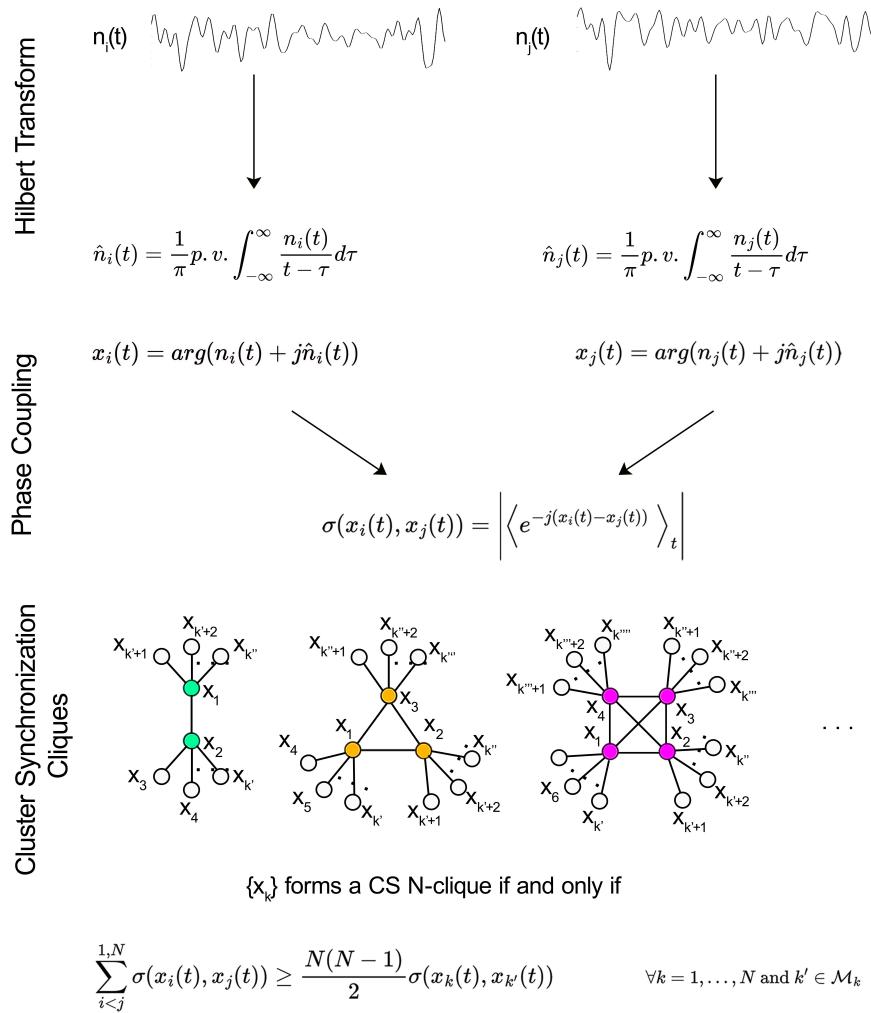


Fig. 25.2

Schematics of the synchronization clustering algorithm for fMRI. Pairs of time series coming from pairs of cerebral ROIs are Hilbert transformed and entered into the phase-locking value calculation. Once all pairs of regions of interest are included in the calculation, the synchronization clustering algorithm is implemented. Figure reproduced from (Gili et al., 2025).

had been obtained for the eleven ROIs included in the study (by spatial averaging the BOLD signal within each ROI at each time point), the synchronization was calculated as follows.

Given the BOLD signals $n_u(t)$ and $n_v(t)$ coming from regions $u, v = 1, \dots, N$ ($N = 11$), their instantaneous phases $\phi_{n_u}(t)$ and $\phi_{n_v}(t)$ can be obtained by means of their Hilbert transforms (Fig. 25.2). The PLV $\sigma(\phi_{n_u}(t), \phi_{n_v}(t))$ is then given by

$$\sigma(\phi_{n_u}(t), \phi_{n_v}(t)) = |\langle e^{-j(\phi_{n_u}(t) - \phi_{n_v}(t))} \rangle_t|, \quad (25.1)$$

where j is the imaginary unit.

To test the statistical significance of the PLVs, a non-parametric permutation test was run by generating surrogate ROI signals randomly rearranged and eventually time-reversed (1,000 permutations). This procedure generates a null distribution that has the same parameters (mean and standard deviation) as the original data and has similar (but not identical) temporal dynamics. This process produced a null distribution of t-statistics that provided the one-tailed p -value estimated using a generalized Pareto distribution for the tail of the permutation distribution (Winkler et al., 2016). Correction for multiple comparisons was provided by thresholding statistical maps at the 95th percentile ($P < 0.05$, FDR) of the maximum t distribution from the permutation (Winkler et al., 2014).

The PLVs were then entered in a $N \times N$ correlation matrix, representing the correlation/synchronization or PLV matrix. Finally, the PLV matrices were averaged across subjects in each experimental condition (resting state, phonemic fluency task, verb generation task). The functional network was then obtained by thresholding the group-averaged correlation matrix.

The cluster synchronization was then obtained from the functional networks by applying the clique synchronization algorithm of Section 23.5.5 defined in (23.4).

The cliques were then detected by applying the standard percolation threshold procedure (Gallos et al., 2012b; Mastrandrea et al., 2023) to define a hierarchy of synchrony clusters according to the order of clique appearance in the percolation process. These clusters are the balanced colorings or fibers that form the input of the MILP of Section 22.5, which is used to infer the connectome.

25.4 Patterns of synchrony during resting state and language task

The results of the RS functional network are shown in Fig. 25.3a, in which the ROIs are colored according to the clusters obtained from the PLV correlation matrix shown in Fig. 25.3b. It is known from the literature that the RS-fMRI functional network is approximately left-right symmetric; i.e., Broca left, and Broca right are activated and synchronized in RS (Teghipco et al., 2016; Seitzman et al., 2019). These results confirm this evidence by demonstrating left-right symmetry in the synchronization of the language ROIs under RS. We find a central CS composed of a pentagonal clique (Li et al., 2019, 2020, 2021) of two bilateral pairs of regions (premotor and Wernicke's area) and the supplementary motor area,

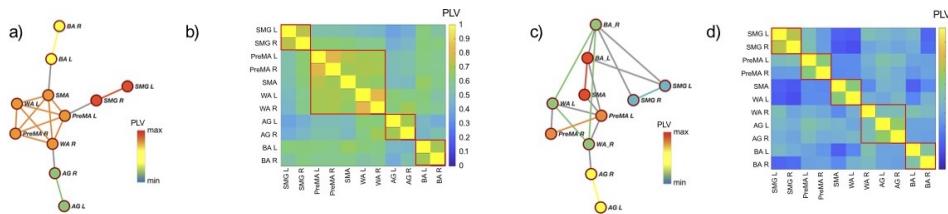


Fig. 25.3

Resting state and language functional network. (a) Cluster synchronization of the functional network obtained under the RS condition. Each color represents a cluster. (b) PLV matrix under RS. The red-lined boxes are visual indicators for the clusters of ROIs. (c) Cluster synchronization of the functional network obtained under task-based language. (d) PLV matrix for the 11 ROIs. Figure reproduced from (Gili et al., 2025).

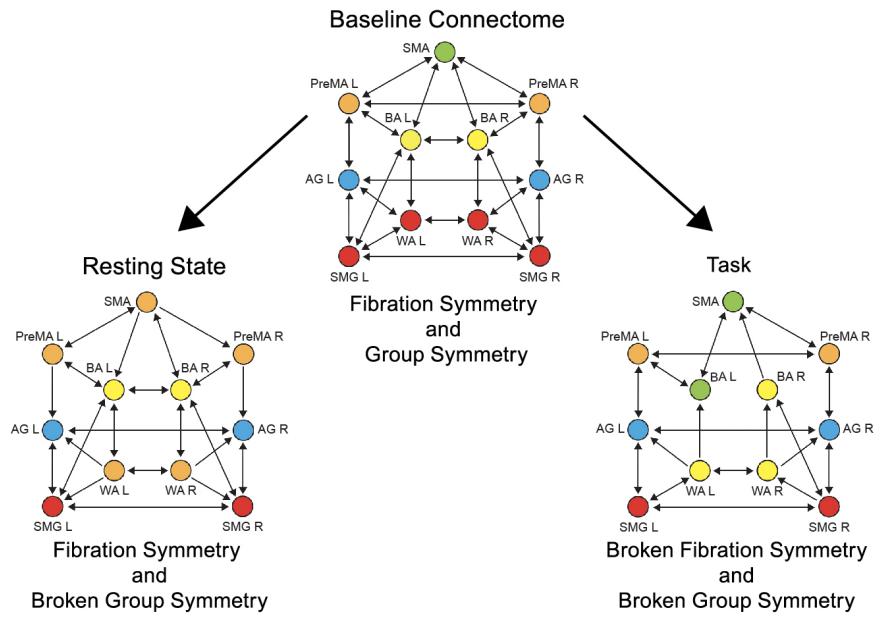
and three CS, each composed of a bilateral pair of regions (supramarginal gyrus, angular gyrus, and Broca's area, Fig. 25.3a).

The same analysis is applied under the phonemic fluency task (Fig. 25.3c, d), showing the clusters formed by SMA and Broca's left to be the most synchronized. The second most synchronized cliques were angular left and angular right. The third most synchronized were premotor left and premotor right, followed by Wernicke's left, Wernicke's right, Broca's right, and finally, the supramarginal left, supramarginal right clique. This clear synchrony pattern is then used as the input of the symmetry-based algorithm of Section 22.5 to infer the pathways of white matter tracts that are responsible for the synchronization.

25.5 From synchronization to the structural network

The active pathways of the brain adjust according to the requirements of the task (Friston, 2011; Park and Friston, 2013). In the present case, given our network of mesoscopic brain regions and all the possible structural connections among them that are identified as the bundles of axons forming the white matter fiber tract connectome, different subsets of these structural connections mediate different functional conditions. Consistent with this idea, a given functional activity (whether at rest or task-specific) is sustained by a specific pathway of the brain structure, which strictly depends on the activity itself. We use this idea to match the patterns of balanced colorings from cluster synchronization with the symmetries of the underlying connectome.

The pipeline to infer the active pathways in the brain is exemplified in Figs. 22.2b and 25.1, and was used to infer the memory engrams in the mice connectome in Chapter 24. The algorithm receives the initial baseline connectome as input, which represents the communication highway. This connectome can be complete or incomplete. In the case of the language function studied here, years of brain analysis have determined the main white matter tracts between the main ROIs considered. The ROIs are anatomically defined in

**Fig. 25.4**

Fibration symmetry breaking in the human brain. (a) Baseline connectome. Structural network between anatomically defined ROIs from the literature. This network has the largest symmetry, which is a global automorphism group. Here, the symmetry group is the same as the symmetry fibration and the five orbits are the same as the five fibers (balanced coloring with five different colors). (b) Rest. Reconstruction of the structural network for the group average of subjects under RS, using the balanced coloring from Fig. 25.3a and the MILP algorithm. The network has only local symmetry, given by fibrations with four fibers (balanced coloring with four different colors), but no global symmetry. (c) Task. The symmetry of (b) is broken by lateralization under the language task. This structural network (reconstructed using the coloring from Fig. 25.3c) shows less symmetry (i.e., broken symmetry with more fibers/five balanced colors shown in the figure) than in (b). Figure reproduced from (Gili et al., 2025).

Section 25.2. The baseline structural connectome is obtained from the literature, and it is shown in Fig. 25.4a.

From this initial connectome (highway system), only a subset of routes is needed to guarantee the existence of the synchronous clusters (fibers/balanced coloring) that satisfy the synchrony pattern observed in Fig. 25.3. The working hypothesis is that the brain connectome is stable over the timescales of the experiments, i.e., no plasticity reorganization of the white matter tracts occur. If the knowledge of the connectome is complete, then any modification we want to induce on it cannot include the addition of new links, so only decimation processes are admitted to lead to the structural network that matches the

coloring of the brain regions to the synchrony coloring. If the knowledge of the connectome is incomplete, then the algorithm also allows the addition of new links. The algorithm from Section 22.5 accounts for both situations.

The symmetry-driven MILP algorithm (Section 22.5) applied to the brain is as follows. It finds the minimal number of links to be removed from the connectome to match the balanced coloring of the graph with the one obtained experimentally from the functional network. The scheme for synchronization-to-symmetry matching in the brain network can be summarized in the following steps, Fig. 25.1:

- For a given set of ROIs, identify the baseline connectome that form the graph of all the permitted structural connections among them;
- By means of the PLV synchronization measure, find the synchrony clusters according to clique synchronization, (23.4). Each cluster is assigned a color such that nodes with the same symmetry have the same color;
- Partition and color nodes according to the fibration/group symmetries,
- Decimate the initial connectome with MILP by removing the minimal number of links until the symmetry coloring of the connectome matches the synchronization coloring.

This optimization problem is solvable with MILP for modestly sized instances.

The algorithm starts with the known structural network containing all the pathways (highway system) shown in Fig. 25.4a. We see that this baseline connectome already displays a remarkably symmetric structure. This network represents the underlying communication highways of the brain involved in language.

25.6 Symmetry breaking in the human language connectome

In general, only a subset of routes is needed to guarantee the existence of synchronous ROIs that satisfy the observed fMRI synchronization. Applying the pipeline in Fig. 25.1 to the baseline connectome of Fig. 25.4a, informed by the fMRI PLV correlation matrix from Fig. 25.3ac, we obtain two reconstructed connectomes for resting and language tasks, shown in Fig. 25.4b,c respectively. That the algorithm can return a solution with minimal removals from the baseline connectome indicates the feasibility of the solution. Otherwise, the algorithm would return either no solution or remove all links since any coloring is balanced when there are no links.

We find that the resting state is a particular realization of the cerebral synchronization pattern, characterized by a defined symmetry that is broken in the rest-to-task transition, determining differential recruitment of structural connections according to its functional state. Consequently, we show that (among the allowed patterns of structural connectivity) synchronization elicits different subsets according to the different functional engagements of the brain, shaping the necessary communication routes.

We find progressive and different symmetry-breaking processes. The baseline language

connectome (highway system) is characterized by the most symmetric configuration. As shown in Fig. 25.4a, the numbers of orbits and fibers are equal, so the symmetry is a group symmetry—and, by definition, also a fibration. That is, the symmetry is a global automorphism. Once the brain dynamics enters its resting state (Fig. 25.4b), this global symmetry is broken, and only the local fibration symmetry survives. Synchronization emerges as a functional condition, being the resting state characterized by only fibration symmetry and a total loss of global group symmetry. This means that the small perturbation represented by brain synchronization, overlapping the static network, suppresses the automorphism but also enhances the biological configuration that allows for activity stability.

Finally, during the execution of the language task, the activity is largely polarized in recruiting areas devoted to language function (Fig. 25.4c). The known lateralization of brain activity during language execution induces a further fibration symmetry breaking observed in the lateralization or breaking of left-right symmetry between Broca's left and Broca's right. This induces five synchronized clusters in the task, as compared to 4 in the resting state, with a concomitant loss of symmetry together with a total loss of global automorphism (one orbit for each ROI). (A larger number of synchronized clusters corresponds to less symmetry.)

The result is significant for understanding the brain in terms of complex systems. Indeed, much of the physical world's complexity emerges from mechanisms of symmetry breaking (Anderson, 1972), and nature's symmetry can be broken in various ways, as discussed in many situations in Part I. One such mechanism is spontaneous symmetry-breaking (Anderson, 1972; Wilczek, 2005, 2016; Weinberg, 1995), which explains all matter and interactions from phase transitions (ferromagnets, superconductivity, etc) to the standard model (Higgs boson, etc). As in the theory of phase transitions in physical systems, we find that the brain switches from a dynamical state (the resting state) to another (the execution of a task), inducing a fibration symmetry breaking of the structural connectivity pathways that sustains synchronization of the brain regions' activity in the different conditions.

25.7 Application of symmetry theory to find essential areas in the brain

A fibration theory of the brain improves our understanding of the design principles of neural circuits, as well as how their structure influences their function. This theory has immediate medical relevance in neurosurgery. For instance, we are working with neurosurgeons and neuroradiologists at Memorial Sloan Kettering Cancer Center in New York City. They routinely perform awake brain surgery during craniotomy for tumor resection on patients with brain tumors. During the operation, neurosurgeons perform direct cortical stimulation of different areas in the brain to search for essential areas for expressive language function (Fig. 25.5).

In the clinical setting, Dr. Holodny and his team of neuroradiologists at MSKCC present the neurosurgeons with fMRI activation maps, which depict the locations of eloquent cortices (such as language and motor areas) adjacent to tumors, which neurosurgeons

use to plan and guide the resection of gliomas and other intracranial masses (Fig. 25.5). Notwithstanding its advantages, fMRI clearly has limitations. One of the main problems facing the pre-operative evaluation of brain tumors by fMRI is that this technology depicts activations of both essential and non-essential functional areas. For a neurosurgeon, this distinction is of paramount importance. In this case, essential language areas are those whose direct, intraoperative stimulation by electrodes leads to speech arrest. Resection of such areas leads to severe language deficits. In contradistinction, non-essential language areas are defined as those whose direct, intraoperative stimulation by electrodes does not lead to speech arrest. Resection of such areas is thought to be safe.

While fMRI technology alone does not allow one to differentiate between core areas, the inference network method used in this chapter and tested in the mouse brain in Chapter 24 allows for this distinction. A fibration theory that can predict these essential areas in the brain contributes to optimizing the decision-making process through which neurosurgeons select which parts of a tumor to resect. Currently, routinely performed preoperative fMRI scans do not discern which areas of fMRI activation represent ‘essential’ language areas. The neurosurgeon, therefore, does not know, in advance of the operation, which areas of the tumor can safely be resected.

A graph theory of the brain has the potential to predict those areas that are essential for language function. Such a result will be unprecedented: prediction/testing of brain ROIs that are ‘essential’ for function via direct cortical stimulation has never before been attempted for the human brain.

Development and testing (in the setting of direct intraoperative cortical stimulation) of theories of organization and response to brain network perturbations should lead to the inference of general principles regarding the network organization of brain circuits and the changes they undergo in pathological states.

Finally, the development of theories of the organization of the connectome should lead to general principles regarding network organization, applicable to areas outside neuroscience, including general information-processing complex systems.

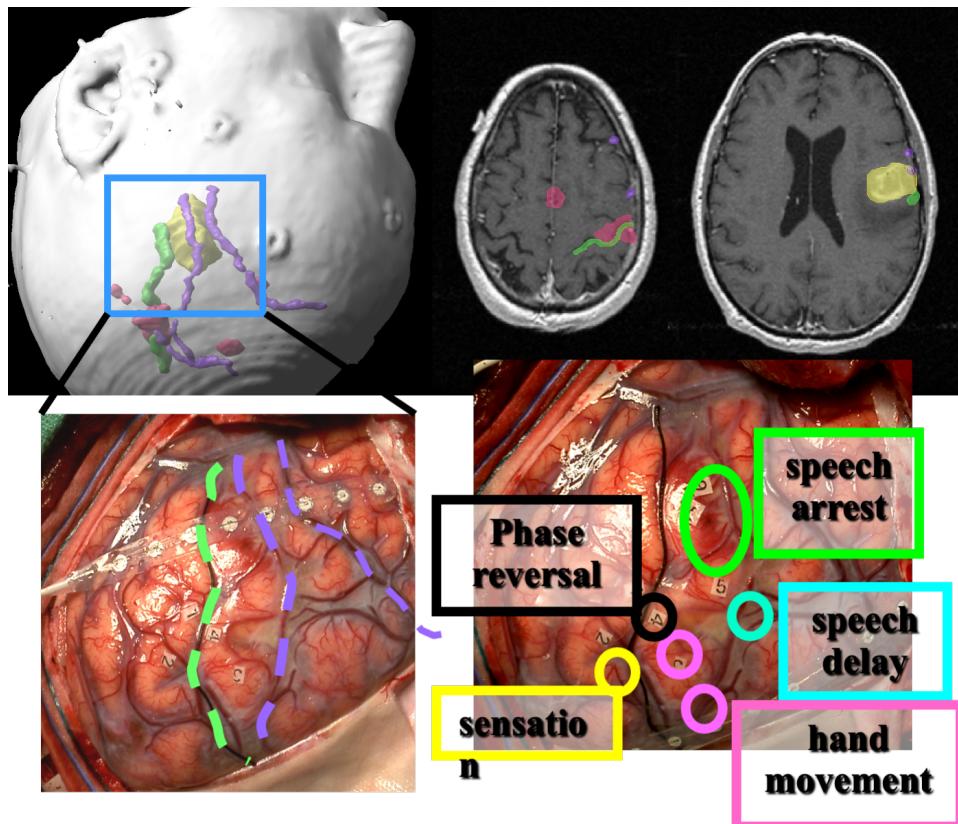


Fig. 25.5

Application of inference graph theory to the brain. Example of how fMRI data is presented to the neurosurgeon for glioma resection. A 3D rendering (top left) in the projection the neurosurgeon will see during the craniotomy portrays the central sulcus (green), the hand motor homunculus (red), the tumor (yellow), and cortical veins (purple). The same information is seen in the 2D images (top right). The fMRI data correlates well with the intraoperative findings. The bottom left shows the exposed brain during the operation, with the green and purple lines corresponding to their counterparts on the 3D MR image. The results of the direct cortical stimulation (bottom right) correspond to the locations predicted by fMRI-based theory. A fibration theory of the brain can identify the essential areas in the brain in advance, which helps the neurosurgeon perform optimized tumor resections with minimal consequences for the patient. Figure courtesy of Dr. Andrei Holodny, MSKCC.

Geometric Deep Learning (GDL) unifies a broad class of machine learning techniques that look at learning from the perspective of symmetries, offering a framework for introducing problem-specific inductive biases in the same spirit as Graph Neural Networks (GNNs) do. Classical formulations of GDL take only global group symmetries into account. Considering local fibration symmetries of graphs, instead, allows us to leverage regularities of more realistic instances. Indeed, GNNs themselves apply the inductive bias of fibration symmetries, and from this observation, we can derive a tighter upper bound for their expressive power. Symmetries in networks let us collapse network nodes, increasing their computational efficiency during inference and training, especially when deep networks are involved. Fibration symmetries synchronize activations during forward propagation in the inference process, while opfibrations synchronize errors during backpropagation in learning. Thus, covering symmetries dominate the deep neural network architecture. These symmetries apply beyond graphs to manifolds, bundles, and grids for the development of models with inductive biases by local symmetries that lead to better generalizations.

26.1 Geometric Deep Learning and the Erlangen Program

The approach of simplifying and ultimately understanding a system through its symmetries finds broader applications than in physics and biology. Recently, symmetries and geometry have become fundamental tools in data science and machine learning. One fundamental application of modern machine learning is to find a dimensional reduction of high-dimensional data to a low-dimensional representation and symmetries have largely been exploited in this endeavor (Bronstein et al., 2021). Machine learning techniques to reduce the dimensionality of data further, include the principal component analysis (Berman et al., 2014), clustering methods, renormalization groups (Meshulam et al., 2019), sloppy models (Transtrum et al., 2015), and others (Stephens et al., 2008, 2010, 2011).

In the machine learning realm, symmetries are transformations applied to data that, while changing the concrete (numerical) representation of the data, do not change the underlying object that the neural network is learning. For example, translating or rotating an image of a cat should not change the classification of the image as a cat. Likewise, permuting the labels of a graph or rotating a molecule in three dimensions should not change its classification. In this case, we are looking at symmetries as a way to constrain the model; in other words,

symmetries within the data structure can be used to develop a neural network that respects the same symmetry. A typical example is translational invariance, built into convolutional neural networks or graph neural networks that are invariant under permutations. Ultimately, symmetry can be considered as a general unifying blueprint for the most popular network architectures, from convolutional neural networks, recurrent neural networks, and graph neural networks to transformers. Following the 1872 Erlangen Program of Felix Klein (Section 11.4.1), the symmetrization of deep neural networks is called Geometric Deep Learning (Bronstein et al., 2021).

26.2 Symmetries in deep neural networks

In various disciplines, such as neuroscience, machine learning, and computer science, researchers are interested in finding effective representations for temporal signals, images, grids, or large graphs to solve tasks in reasoning, prediction, or learning (Baker et al., 2022; Poldrack, 2021). Of course, one way to find effective representations is to identify *symmetries* in the data. In this context, symmetry is a transformation that preserves a particular property of the data (Bronstein et al., 2021).

In machine learning, symmetries are crucial for successful neural network architectures (Higgins et al., 2022). For instance, Convolutional Neural Networks (CNNs) exploit translational symmetry (Alzubaidi et al., 2021), which is useful in image processing where the position of an object can change without changing its appearance. Other examples include Transformers (Lin et al., 2021) and Graph Neural Networks (GNNs) (Zhou et al., 2020), which exhibit permutational symmetry, ensuring that the output remains unchanged regardless of the order of items in the input. Incorporating symmetries enhances data efficiency and reduces learning complexity (Higgins et al., 2022), constraining the search in the model space to models that respect the inherent symmetries of the data. Geometric Deep Learning (GDL) proposes a taxonomy of deep neural network (DNN) architectures based on their symmetries and the symmetries of the data that they deal with (Bronstein et al., 2021).

Here, we focus on graph symmetries, particularly graphs with dynamic states, such as the ones used in GNNs. The predominant symmetry explored in the GDL literature is that of a graph automorphism (Bronstein et al., 2021; Harary, 1993; Higgins et al., 2022) (Definition 4.2), where node permutations are required to preserve the *global* structure of the graph. While useful, we already know that preserving global structure is often too restrictive for real-world graphs. In this book, we are mainly interested in fibration symmetries,, but in Section 6.6, we also discussed two other notions of symmetry: op-fibrations and coverings. All these notions do not preserve the global structure but maintain some graph dynamics. Fibration symmetries, in particular, are transformations between directed graphs that preserve the input trees of nodes, thus preserving local dynamics. Nodes with isomorphic input trees can synchronize, meaning that they share the same dynamics; we have seen this many times already in many different contexts.

In computational graph networks, symmetries influence both the expressive power and the dynamics of computations in inference (fibrations) and learning (fibrations). Fibration

symmetries also offer a natural compression method for graph-structured inputs to such networks. For instance, synchronized genes in a gene regulatory network (Barbuti et al., 2020) can be treated as a single entity for dynamic analysis, improving the scalability of memory and computations in GNNs (Zhou et al., 2020). Unlike other compression methods (Deng et al., 2020; Liang et al., 2021), fibration symmetries ensure that the compressed graph correctly simulates the computations on the original (uncompressed) graph. Previous heuristic compression methods for GNNs, based on neighborhoods (Bollen et al., 2023), show comparable learning performance. Since fibration symmetries preserve the dynamics of networks without the need to preserve their global structure, this framework can be applied across a wide range of networks in machine learning.

In the rest of this chapter, we discuss an upper bound on the expressive power of GNNs based on fibration symmetries (Section 26.3). Additionally, we demonstrate the utility of identifying fibration symmetries within a dataset composed of graphs (Section 26.4) for optimizing computations in GNNs while preserving performance (Section 26.4.1). Upon observing the emergence of synchronized nodes in Deep Neural Networks (DNNs) trained via gradient descent (Section 26.5), we define a new training algorithm that exploits symmetries to optimize DNN architectures (Section 26.6). Finally, we show in Section 26.7 that while fibrations describe the synchronous activations in the inference process, opfibrations, with an emphasis on the output trees, dominate the learning process via backpropagation of error signals. Thus, coverings, which preserve both input and output trees of a given node, are relevant for deep neural architectures. Most of the material presented here summarizes (Velarde et al., 2025).

26.3 Upper bound on the expressive power of GNNs

In this section, we show that the expressive power of GNNs is intimately related to the symmetries of the graph and extend previous theoretical work to the case of fibration symmetries to obtain tighter bounds on the expressive power of GNNs.

In Section 13.2 we mentioned the Weisfeiler–Lehman test, or WL-test. The WL-test is, in fact, a family of GITs (Graph Isomorphism Tests), which can be used as heuristics to determine whether two graphs are isomorphic. More precisely, a GIT is a polynomial-time algorithm that can be used to answer *negatively* the question of whether two given graphs are isomorphic: negative answers are always correct, whereas positive answers by a GIT mean that the test could not determine whether the graphs are isomorphic.

There exists a full family of WL-tests, one for each value of a parameter k : the lower level of this family is called 1-WL and is the only one mentioned here. The usual formulation of 1-WL applies only to undirected graphs, and for those graphs, it coincides with color refinement. When adapting it to directed graphs, we must decide how to run color refinement—whether with incoming or outgoing arcs. If we consider the Weisfeiler–Lehman test extended with incoming arcs, we can reformulate and extend the theorems of (Xu et al., 2018) in terms of fibration symmetries:

Theorem 26.1 (Velarde et al., 2025) (a) Let G_1 and G_2 be graphs with minimal fibrations $\phi_i : G_i \rightarrow B_i$ ($i = 1, 2$) and $B_1 \not\cong B_2$. If a graph neural network $\mathcal{A} : \mathcal{X} \rightarrow \mathbb{R}^d$ maps G_1 and G_2 to different representations, then the Weisfeiler–Lehman test (see Section 13.2) also decides that G_1 and G_2 have different minimal bases. (b) Let $\mathcal{A} : \mathcal{X} \rightarrow \mathbb{R}^d$ be a GNN. With a sufficient number of GNN layers, \mathcal{A} maps to different representations of any graphs G_1 and G_2 that the Weisfeiler–Lehman test decides have different minimal bases, provided that \mathcal{A} aggregates and updates node features iteratively with

$$h_v^{(k)} = \gamma \left(h_v^{(k-1)}, f(\{\{h_u^{(k-1)}, u \in N(-, v)\}\}) \right),$$

where the functions f and γ are injective, and the graph-level readout of \mathcal{A} is injective.

This theorem shows that the expressive power of GNNs is precisely constrained by the expressive power of the Weisfeiler–Lehman test. We emphasize that for undirected graphs, input trees, and output trees are the same, therefore, in the undirected case, these results are identical to those of (Xu et al., 2018).

We can generalize this statement to incorporate the various levels of symmetry (i.e., isomorphisms/automorphisms, coverings, fibrations) defined in Section 6.6. Each level of symmetry defines an inductive bias and a partition of the Space of Graphs. Figure 26.1 shows, in a nutshell, the relations between levels of symmetry and expressivity. Intuitively, there are more graphs with the same fibration base than there are isomorphic graphs. Thus, there are more equivalence classes of graphs for stronger symmetries. For each inductive bias, we calculate the maximum expressive power and find the order (Velarde et al., 2025) (Fig. 26.1b):

$$E_{F_{fib}} \leq E_{F_{cov}} \leq E_{F_{iso}}. \quad (26.1)$$

Intuitively, each inductive bias incorporates information about the symmetry utilized: the map F does not distinguish between graphs in the same class (Fig. 26.1c). In Fig. 26.1d, we show how expressive power and inductive bias vary with symmetries. Global symmetries (e.g., isomorphisms/automorphisms) are stronger and henceforth induce smaller inductive biases and larger expressive power. For instance, the strictest equivalence relation is graph equality; in this case, any F satisfies the bias condition, i.e., there is no bias applied to the mapping at all. However, the least strict equivalence relation is that all graphs are equivalent; in this scenario, the mapping F must be constant—a very strong inductive bias.

Another way to visualize the expressive power is through the volume of the set of mappings that satisfy a certain bias condition (see Fig. 26.1e). The set of mappings with inductive bias induced by fibrations (yellow area) is a subset of those with inductive bias induced by coverings (red area). This latter set is also a subset of the functions with inductive bias induced by automorphisms (blue area). GNNs are particular functions that satisfy the bias induced by fibrations (green line inside the yellow area), since they depend solely on the in-neighborhoods.

These statements are about the *expressive power* of a GNN. A different and equally important question is about the *performance* of a GNN, i.e., how well does a model learn to perform a particular input-output mapping for a real-world dataset? We discuss this aspect in the rest of this chapter.

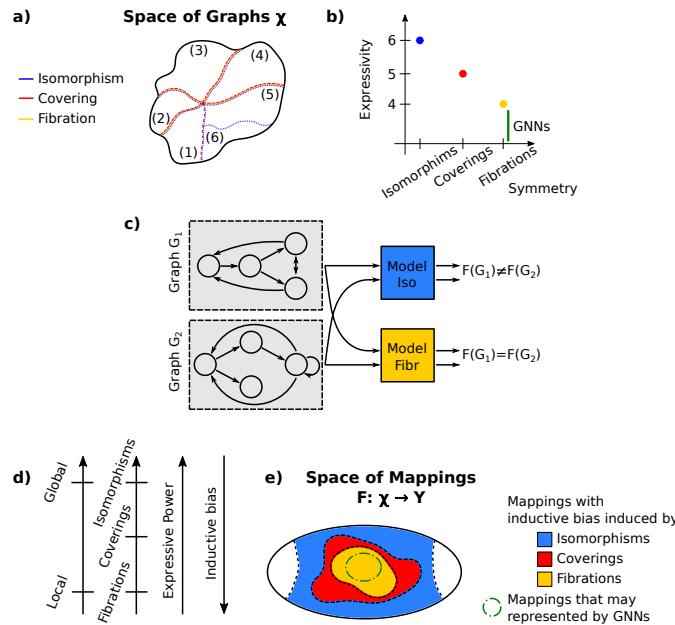
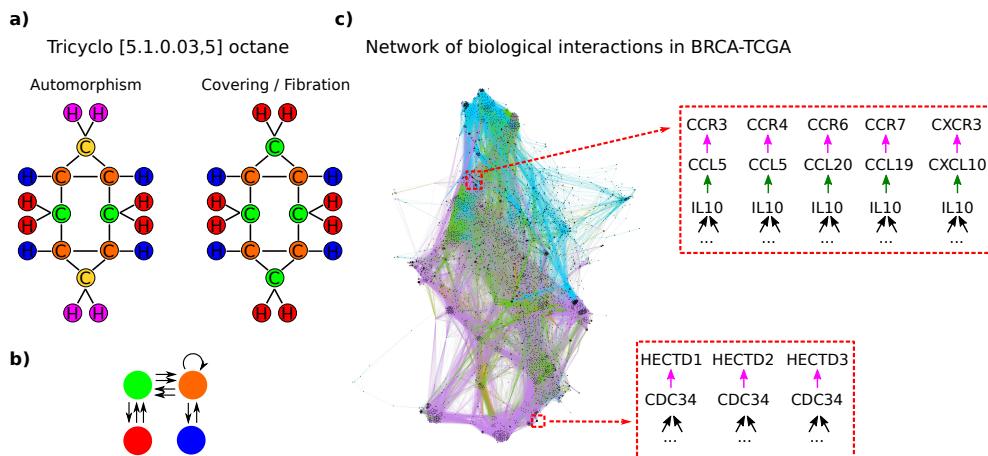


Fig. 26.1

Symmetries, inductive bias and expressive power. (a) For each level of symmetries (i.e., fibrations, coverings, automorphisms), there is a partition of the space of graphs X (colored dashed lines). The partition induced by isomorphisms (e.g., 6 classes) is finer than the partition induced by coverings (e.g., 5 classes); and the latter is finer than that induced by the fibrations (e.g., 4 classes). (b) Functions with inductive bias induced by isomorphisms have greater expressive power than those that satisfy the inductive bias induced by local isomorphisms (i.e., coverings) or induced by local in-isomorphisms (i.e., fibrations). (c) The graphs G_1 and G_2 are not isomorphic, but they have the same fibration base. A map F with inductive bias induced by isomorphisms (blue box) returns different outputs for the graphs (i.e. $F(G_1) \neq F(G_2)$). On the other hand, a map F with inductive bias induced by fibrations (yellow box) returns the same output for both graphs (i.e., $F(G_1) = F(G_2)$). (d) Isomorphisms are global symmetries, while coverings and fibrations are local symmetries. Isomorphisms, coverings, and fibrations induce different inductive biases, which increase when the symmetry is weaker. This implies that expressive power is less for models that use weaker symmetries. (e) In the Space of Mappings, there is a subspace of maps that satisfies the inductive bias induced by isomorphisms (blue area), coverings (red area), and fibrations (yellow area). GNNs are a subset of the maps with bias induced by fibrations (green line). Figure reproduced from (Velarde et al., 2025).

**Fig. 26.2**

Fibration symmetries in datasets. Examples of the fibration symmetries observed in the graphs of QM9 and BRCA-TCGA datasets. (a) Tricyclo [5.1.0.03,5] octane is a molecule composed of eight carbon atoms and twelve hydrogen atoms. The partition induced by automorphisms consists of six orbits, while the partition induced by fibrations consists of four fibers. One fiber consists of 4 hydrogen atoms (blue circles), one fiber of 8 hydrogen atoms (red circles), and two fibers of 4 carbon atoms (green and orange circles) (b) Fibration base of Tricyclo octane. (c) The biological interaction network in BRCA-TCGA consists of 9,288 genes interacting in various ways represented by arrows of different colors. The genes CCR3, CCR4, CCR6, CCR7, and CXCR3 are in the same fiber because they have isomorphic input trees. The genes CCL5, CCL20, CCL19, and CXCL10 (resp. HECTD1, HECTD2, and HECTD3) are in the same fiber because they receive information from the same gene IL10 (resp. CDC34). Figure reproduced from (Velarde et al., 2025).

26.4 Fibration symmetries in graph-structured datasets

Here, we report fibration symmetries and the corresponding base graph for two real-world datasets (Fig. 26.2), namely, the Quantum Mechanic 9 (QM9) dataset, which provides quantum chemical properties for a large number of small organic molecules, and The Cancer Genome Atlas Breast Cancer (TCGA-BRCA) dataset, which includes gene expression and survival time of individual patients with breast cancer. Both datasets consist of graph-structured samples. We calculate the minimal fibration base B of each graph G using the algorithm of Section 13.7.

Some examples of the graphs involved and their corresponding partitions are shown in Fig. 26.2. From the QM9 dataset, we show the structure of Tricyclo [5.1.0.03,5] octane. The graph structure of Cyclopropylmethanol in Fig. 26.2a has 4 fibers (see panel Covering/Fibration). To maintain the identity of the atoms within a fiber, there cannot be two different

types of atoms. The partitions induced by the fibration and covering symmetries coincide since the graph is undirected. However, they are not necessarily the same as the partition induced by automorphisms. For this molecule (Fig. 26.2a), there are six orbits derived from the horizontal and vertical reflection symmetries. For each molecule, we build the fibration base; e.g., in Fig. 26.2b, we show the base of Tricyclo octane. The compression factor (i.e., the number of nodes in the base divided by the number of nodes in the molecule) for Tricyclo octane is $4/20 = 0.2$. This value depends on the molecular structure. The average compression factor computed in (Velarde et al., 2025) across all samples in the QM9 dataset is $\approx 74\%$.

In both datasets, there are structures with different types of interactions. For example, molecules have simple, double, and/or triple bonds. In the biological interaction network of the TCGA-BRCA dataset, some interactions represent the expression control of a gene (green arrows), activation/inhibition of protein complex components (pink arrows), or reaction control of a protein (light blue arrows). This dataset includes only one graph of 9,288 genes with a compression factor of $\approx 53\%$. In Fig. 26.2c, we show examples of fibers in the network.

26.4.1 Training GNNs with minimal fibration base

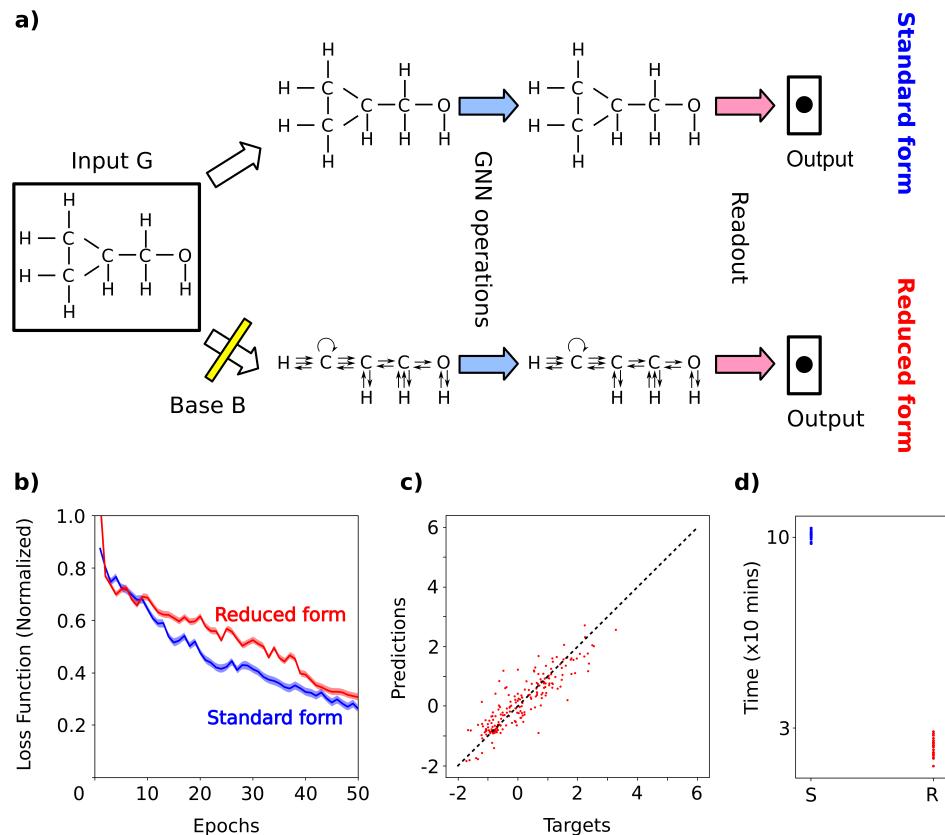
Such training maintains performance. We can consider the following prediction tasks related to the dataset we described:

1. Predict the dipole moment of each organic molecule based on its structure.
2. Predict the overall survival time of each patient based on their gene expression profile.

For both tasks, we can train a GNN network using the QM9 and BRCA-TCGA datasets, respectively. Figure 26.3b-d show the results for **T1**.

The evolution of the loss function over the training steps for both forms is similar, see Fig. 26.3b. In particular, after 50 epochs, the difference is only 4%. In Fig. 26.3c, we show the predictions for the GNNs versus the actual dipole moment values for the molecules in the test set. The loss function is calculated as the mean squared error of these values (i.e., distance to the diagonal). A relevant advantage of the reduced form is its execution time, which is 4 times shorter than the time for the standard form (see Fig. 26.3d). An important result is that the reduction in the execution time is much greater than the reduction in the samples. The results for **T2** are consistent with the behavior observed in Fig. 26.3.

Given that GNNs satisfy bias induced by fibrations, it is expected that the learning process should remain invariant under fibration symmetry (i.e., graph compression from G to B). The results we presented confirm empirically that the performance of a GNN is equivalent when trained on either the original or compressed graphs of real-world datasets. In other words, we leveraged that every (learning) procedure executed on the base replicates the same procedure executed on the original graph network. Moreover, this approach improves computational efficiency without compromising learning performance.

**Fig. 26.3**

Standard and Reduced form of GNNs. (a) The structure of a GNN and the number of operations are derived from the structure of the input graph G (Standard form). Finding the fibration base B for the input graph G lets us reduce the number of operations in GNNs (Reduced form). (b) Loss function during the training stage for the task T1. (c) Predictions of the network vs. dipole moment of the molecules. (d) Comparison of the execution times for ‘standard form’ and ‘reduced form’ for task T1. Figure reproduced from (Velarde et al., 2025).

26.5 Fibration symmetries and synchronization in training deep neural networks

Given the results presented above, we can ask whether fibration symmetries also play a role in conventional DNNs, such as the Multi-Layer Perceptron (MLP) (Fig. 26.4a), which has been taken as a prototype for modeling the brain through neural networks. A common

observation in the brain is oscillatory activity, where the activity of neurons changes in unison, a form of synchronization of the neurons themselves (Thut et al., 2012).

When first initialized, MLPs have a random weight on the graph structure, giving rise to diverse ‘activation’ in each neuron. We hypothesize that, during learning, the weights adjust themselves to provide synchronized network activity, i.e., two or more nodes converge during learning to have the same activation values for all input samples. To test this, we analyze the activity of the nodes in MLPs that were trained and achieved an accuracy of 97.67%, 90.58%, and 88.42% (Fig. 26.4b) for MNIST, KMNIST, and Fashion datasets, respectively.

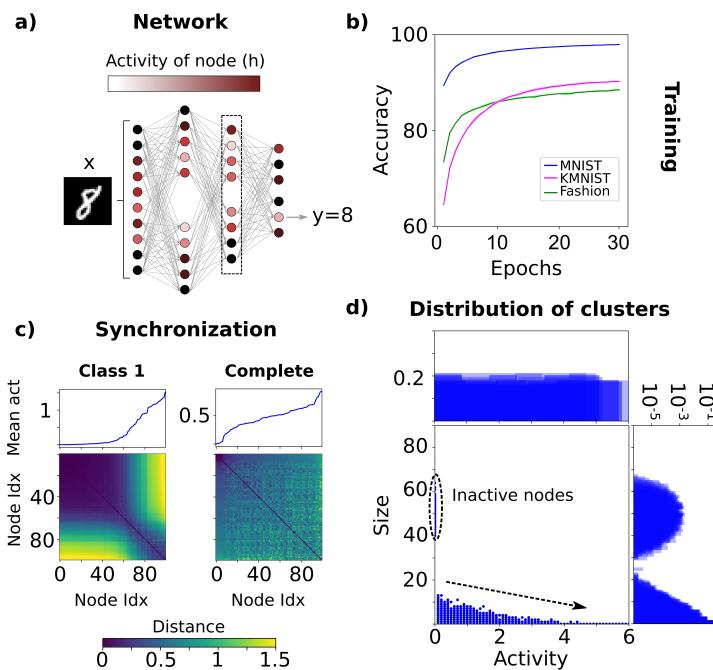


Fig. 26.4

Synchrony in MLP. (a) An MLP trained to classify images (e.g. digits in MNIST dataset). (b) Evolution of MLP accuracy during the training stage for different datasets: MNIST, KMNIST, and Fashion. (c) After the training stage, clusters of synchronized nodes are detected. The clusters depend on the data \mathcal{S} . The results presented in the left (resp. right) column correspond to cases where \mathcal{S} is the subset of samples from class 1 (resp. from all classes) within the test set. The distance matrix between nodes Λ (see definition in Methods) is represented by a color map. The node indices are sorted by mean activity across all samples in \mathcal{S} (see Mean act). (d) In the Size vs. Activity plane, we show the distribution of cluster size and activity across input images (i.e., \mathcal{S} is a single image). In the upper (resp. right) panel, the marginal distribution of cluster activities (resp. sizes) is plotted. Figure reproduced from (Velarde et al., 2025).

In Fig. 26.4c, we plot the mean activity of nodes in the second hidden layer after

the training process (see panel ‘Synchronization’). The mean activity is calculated as the average of the activity of each node across different samples. The node indices are sorted by mean activity. When samples are from the same class (e.g., samples of class 1 in MNIST), we observe large, well-defined clusters of synchronized nodes compared to when synchronization is determined across the entire dataset (see color map in Fig. 26.4c). Moreover, at the beginning of training, the activity values h_i are of the order of ε_l , suggesting the presence of only one trivial cluster in the distance matrix (data not shown).

Similarly to previous cases, clusters (or fibers) of different sizes and associated activity values appear for each sample. In other words, we obtain a distribution of fibers across the samples of the dataset. In Fig. 26.4d, we show the distributions of the size and activity of the fibers across input images. The marginal activity distribution is uniform, while the marginal size distribution is bimodal. One mode is associated with fibers of inactive nodes ($h_i = 0$) whose typical size is around 40–60% of the total nodes. The zero activity of these nodes arises from the ReLU activation function applied to non-positive inputs. The rest of the nodes cluster such that clusters with higher activity are smaller.

26.6 Application of fibration symmetries to gradient descent

We can also apply the symmetry properties of fibrations to modify the gradient descent method, which is used to minimize the loss function during the training stage. More precisely, with this alternative version called *Fibration-Gradient Descent* —*FB-GradDesc* (Velarde et al., 2025)— we construct a smaller network B (the base) composed of fibers from the original network G (see Fig. 26.5a, b), and update the parameters of B using the information from the parameters of G . The way we update the parameters ensures that the input information of each node is preserved, as required for fibration symmetries.

The alternative algorithm for gradient descent using fibration symmetries depends on the set S of samples used, and on a threshold ε (see color bar in Fig. 26.5b): ε is used to decide when two nodes should be considered as having the same activity. The choice of ε is critical: for FB-GradDesc, we use $S = \mathcal{D}_{\text{training}}$ and determine the optimal threshold ε . Additionally, we can use a time constant T , which represents the number of epochs required to detect the clusters and construct the base B . In other words, for every T epochs, FB-GradDesc calculates the fibers using the symmetry criterion and reduces the size of G .

The definition of symmetry imposes stronger constraints as $\varepsilon \rightarrow 0$. In other words, the criterion for symmetry becomes more stringent, and it becomes more challenging to find non-trivial clusters (in the limit, only nodes with the same exact activity are considered to be symmetrical). Conversely, T should be sufficiently large for clusters to emerge. These observations are reflected in the results shown in Figure 26.5c. The size and performance of the network decrease as a function of ε and increase as a function of T . The original gradient descent method is the limit case when $T \rightarrow \infty$ and $\varepsilon \rightarrow 0$.

For instance, if we apply this technique to the classical MNIST example with $T = 5$ (red curve) and $\varepsilon = 5 \times 10^{-2}$, the performance of the network trained with FB-GradDesc

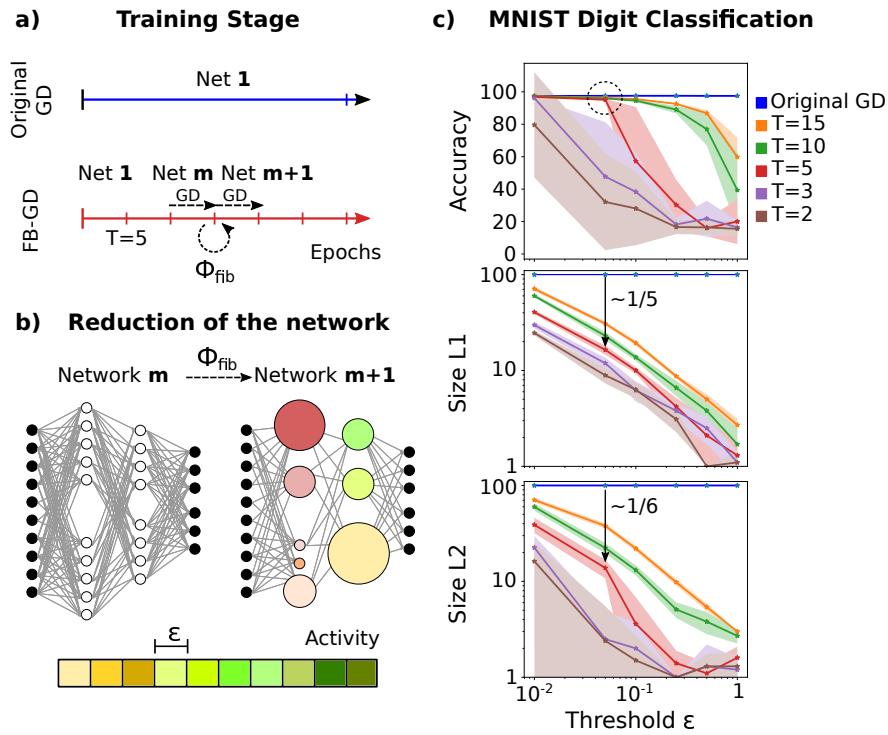


Fig. 26.5

Fibration Gradient Descent (FB-GD). (a) Usually, during the training stage of an MLP via gradient descent, the network architecture remains unchanged over time (see Net 1 - Original DG). For FB-GD, both the connection weights and the network structure are updated (see FD-GB). Every T epochs, clusters of nodes are detected and collapsed into a single node, similarly to constructing the bases (Net $m + 1$) of a graph (Net m) through fibration symmetries Φ_{fib} . (b) Each cluster (colored circles) is associated with an activity interval of size ε (see color bar). The updated equation for the parameters of the network $m + 1$ is designed to be compatible with the update equation for the previous network m . (c) The final size of the network (see Size L1/L2) and its performance (see Accuracy) depend on ε and T . In the limit, $\varepsilon \rightarrow 0$ and $T \rightarrow \infty$, the results of FB-GD coincide with those of the original gradient descent. Figure reproduced from (Velarde et al., 2025).

(accuracy = 95.73%) remains within $\pm 2\%$ of the original network and gradient descent (97.67%). However, with FB-GradDesc, the network could be compressed by a factor of 5 and 6 in the hidden layers, respectively.

26.7 Fibration, opfibrations, and coverings in DNNs

We recall that (op)fibration symmetries preserve (output)input trees (op stands for opposite in category theory), maintaining the local structure of connections: the input tree represents the hierarchical structure of connections leading into the node, while the output tree represents the structure of connections propagating outward from the node.

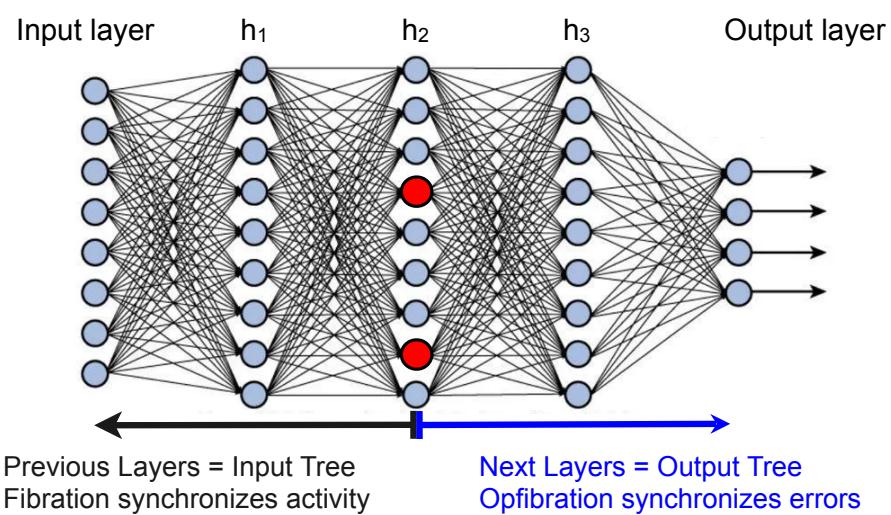
Consider a deep feedforward neural network with multiple layers (Fig. 26.6). Fibration symmetries imply the existence of nodes with isomorphic input trees (black arrow), which synchronize their activations during forward propagation (i.e., inference). Similarly, opfibration symmetries imply that nodes with isomorphic output trees (blue arrow) synchronize their error signals during backpropagation (i.e., learning).

There are cases where nodes exhibit both isomorphic input trees and isomorphic output trees. These scenarios imply the presence of covering symmetries, which lead to the synchronization of both the nodes' activations h and their error signals δ . For instance, the nodes in red in Fig. 26.6 may have isomorphic input and output trees forming a covering of the DNN. This synchronization property further extends to the weight updates during training, dW , as the gradient-based weight adjustment:

$$dW = \eta h \delta \quad (26.2)$$

where η is the learning rate.

These results show that the study of graph symmetries in DNNs opens the door to developing more efficient training algorithms. By identifying symmetric structures, it becomes possible to reduce the parameter space, leading to faster convergence and lower resource consumption. Additionally, symmetries can provide insights into the robustness and interpretability of neural networks, as they often reflect underlying regularities in the data of the problem being solved.

**Fig. 26.6**

Fibration, opfibrations, and covering in DNN. Deep feedforward neural network with $N = 3$ hidden layers. For node i in a hidden layer L , its input tree is defined as the set of all paths from the input layer to node i . This input tree encompasses the layers $L - 1, L - 2, \dots, 1$, and the input layer. On the other hand, the output tree of node i consists of all paths from node i to the output layer, including the layers $L + 1, L + 2, \dots, N$, and the output layer. These paths are relevant in training and inference in DNNs.

Fundamental laws of physics are conceptually simple based on a high degree of symmetry, yet their consequences are enormously rich. We have explored whether the same reasoning can be applied to understand biological systems. Although the group symmetries of physics have some biological applications, the more relaxed concept of fibration symmetry is more widely applicable, and provides insights into fundamental issues in biology. In the first section of this final chapter we summarize some of the key messages of the book, and in the second section we sketch possible topics for future applications.

27.1 Overview

In this section we summarize some of the more important conclusions to be drawn from the ideas presented in this book.

Global symmetries, formalized in terms of groups, have been enormously successful as a basis for fundamental physics. They also have some applications to biology. However, more general and more flexible notions of symmetry are better suited to biology, especially in the age of Big Data Omics.

We learned that these symmetries are organized in a hierarchy, progressing from less strict to more strict:

Box 27.1

Hierarchy of Symmetries

homomorphisms → fibrations or opfibrations → coverings → automorphisms

describing successively different domains of nature:

Box 27.2

Application Domains of Symmetries

biology (genes/brain) → artificial intelligence → physics and geometry

There is an elegant and substantial body of mathematical theory connected with fibration symmetries. It includes fundamental concepts such as equitable partitions, also called balanced colorings, which are determined by the fibers of a fibration. It provides general methods for analyzing models, computing stabilities, and studying bifurcations. These

methods fit into a coherent framework rather than just being the result of one-off numerical calculations on some specific model.

Fibration symmetries in biological networks provide information about biological function and its relation to the form (topology) of the network. However, addressing this question is challenging because biological data are incomplete. We have discussed the problem of incomplete/missing data in biological networks by developing a reconstruction optimization algorithm for the network that reproduces the observed synchronization.

Fibration symmetries permit simple routes to evolutionary change. Under selection pressure for coexpression, fibration symmetries may come into existence when genes are duplicated by lifting processes and are later conserved. This is just a possibility, but we studied it empirically.

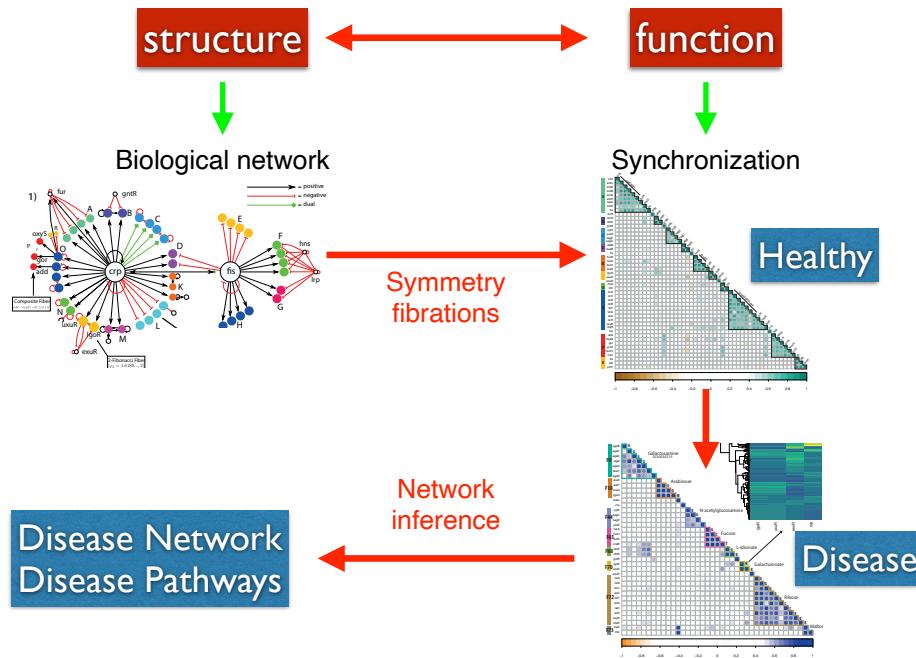
Through evolutionary rewiring and selection, regulatory networks can be modified to perform signal processing tasks. Genetic circuits that are performant, cheap, and robust may evolve many times. Some can be identified statistically as network motifs. However, motifs determined by statistical over-representation should be treated with caution because the dynamics of a motif considered in isolation may differ from its dynamics when embedded in the full network. In contrast, fibration building blocks can be determined directly, without the need for a null model, even if they appear only once in a network. Dynamics arising from fibration symmetries—in particular synchrony—occur in the full network, not just in the motif. Genetic fibers provide insights into coexpression that could not have been achieved by alternative theories, e.g., network motifs, modules, etc.

Our approach bridges the gap between structure and synchronization and can be used to further assess the functionality of the biological network via perturbations to its structure. The concept of fibration symmetry suggests a number of potential experiments. For example, the predictions of the theory can readily be checked by controlled gene knockout experiments designed to break the symmetries of the fiber. Rather than the ‘trial and error’ protocols currently employed, we envisage theoretically guided protocols of gene mutations designed to break specific symmetries of the fibers. This idea also suggests numerous investigations in synthetic genetic circuits which can be designed in a systematic way to build a living GRN made of fibration symmetries and their broken symmetries.

Going beyond the symmetries found in TRNs, the fibration framework can be adapted to include any type of regulation, or any type of interaction among biomolecules. This includes protein-protein interactions, metabolic reactions, and metabolic effects on transcription, as well as signaling pathways that control cell behavior. We envisage an integrative model of gene regulations and metabolism that includes all interactions and molecules that contribute to cellular processes that could be valid across species from bacteria to eukaryotes.

27.2 Outlook

To round off the book, we focus on two of the most promising applications of fibration theory: the understanding of disease and AI. We discuss how fibration theory could help to predict disease pathways in the regulatory networks, to assist designing therapeutics,

**Fig. 27.1**

Finding disease pathways with fibrations. Inferring disease pathways using fibrations involves analyzing gene/neural network synchronization. A healthy genetic or neural network can be established through synchronization experiments and inference optimization algorithms developed in Section 22.5. This healthy network serves as a baseline to infer a disease network by feeding the differential synchronization patterns in the disease state into the inference algorithm.

and to design novel artificial neural architectures that might inspire further work in these directions. We make no attempt at complete coverage, and anticipate that unexpected new developments will introduce new directions, not mentioned here. The aim is to try to justify the view—which, of course we hold, rightly or wrongly—that this book is just a beginning.

Building the AI of the future. The application of fibrations to artificial neural networks opens the door to developing more efficient architectures and novel training algorithms. Symmetries facilitate the dimensional reduction of parameter space, leading to faster convergence and lower resource consumption. Furthermore, these symmetries can enhance the robustness and interpretability of neural networks, as they often reflect underlying regularities in the data associated with the problem being solved. Along the path of geometric deep learning, fibration symmetries and covering spaces represent an extended blueprint for the development of the novel neural architectures of the future.

Uncovering disease pathways. Symmetry theory is also significant for the understanding, diagnosis, and development of treatment strategies for disorders thought to be due

to damage to connectivity. In the case of brain diseases, this includes, e.g., Alzheimer's disease, ADHD, strokes, traumatic brain injury, and schizophrenia (van den Heuvel and Sporns, 2019).

A primary application of the synchronization-driven inference method of Section 22.5 is the understanding of disease pathways. The inference of pathways from dynamical data on healthy subjects can be extended to neurological or psychiatric conditions as well as genetic ones, allowing the identification of differential disease pathways, leading to an understanding of the disease, establishing the diagnosis, and ameliorating the consequences (see Fig. 27.1).

Moreover, the method is beneficial for drug development by targeting the inferred structural network of a specific disease onto a healthy one. The treatment of neurological and psychiatric diseases through invasive intervention (surgery) or non-invasive intervention (electric/magnetic stimulation) will benefit from the identification of the patterns of symmetry and synchronization, and their breaking processes, to reduce side effects or to optimize the effectiveness of the application.

Understanding the organization and response to perturbations of genetic and brain networks leads to general principles regarding network organization of biological circuits and the changes they undergo in pathological states. This occurs because desynchronization/-gain of synchronization can be seen as a pathological state, e.g., cancer. The analyses have the potential to predict the circuits and pathways responsible for pathologies, which can subsequently be targeted to restore the healthy state.

Such advances are crucial for developing more accurate and dynamically responsive neural and genetic models, which can be pivotal in both basic research and clinical applications, such as designing targeted treatments for neurological disorders involving dysregulated neural synchronization in complex neural systems.

Furthermore, exploring the impact of structural variations on the synchronization in disease models could open new therapeutic avenues, particularly for neurological disorders where dysregulation of synchronized activity is evident.

We hope this book will inspire these exciting developments in the future.

A

Appendix A Software, code, and data

We describe a list of software packages used in each chapter. This is handy for those readers who want to delve into the application of fibrations to the study networks in various fields. The general repositories to reproduce all the results in the book and to perform new fibration analyses are available at <https://GitHub.com/MakseLab> and <https://osf.io/4ern8>. All URLs were accessible as of February 2025.

A.1 List of software implementations, codes, and datasets used in this book

1. Codes and data for Chapters 14 and 15

The software for Chapters 14 and 15 contains the primary codes used to conduct a fibration symmetry analysis of networks. This analysis involves calculating balanced colorings and constructing simple building blocks of fibers. These codes are essential for any fibration analysis, as they systematically take a network, identify all fibers, and categorize the building blocks into distinct groups of simple components. The software addresses the forward problem by starting with a graph and identifying and classifying all building blocks based on their symmetries. Additionally, the codes in Chapter 18 will enhance this study by focusing on symmetry-breaking building blocks. Ultimately, the codes for Chapter 19 will integrate all these elements to perform a dimensional reduction, resulting in the minimal fibration base of the network, akin to a 'minimal genome'.

- Fibration symmetries and fibration building blocks— Algorithms developed by Morone et al. (2020) based on balanced coloring algorithms by Kamei and Cock (2013a) and Cardon and Crochemore (1982). Available at <https://GitHub.com/MakseLab/FibrationSymmetries>, and <https://osf.io/4z2hb>.
- Fibration symmetries and fibration building block— Data and analyzed networks developed by Morone et al. (2020). Available at <https://osf.io/z793h> and <https://osf.io/pwe56>, respectively. This includes a list of all simple building blocks found in the TRN networks across species.
- Fast Fiber Partition—Julia implementation of the Fast Fibration algorithm developed by Monteiro et al. (2022). Available at <https://GitHub.com/MakseLab/FastFibration>.

- Automorphisms— McKay’s *Nauty* algorithm (McKay, 1990) to find automorphisms is available at <https://pallini.di.uniroma1.it>. Pynauty, a Python library based on Nauty to work with graph automorphisms, is available at <https://GitHub.com/pdobsan/pynauty>.

2. Codes and data for Chapter 16

These codes systematically classify the complex building blocks described in Chapter 16.

- The codes and data to find the complex building blocks defined in Chapter 16 developed by Álvarez-García et al. (2025b) are available at <https://github.com/MakseLab/Symmetries-in-Metabolic-Networks-of-E.-coli> and <https://osf.io/2ntr3>. They are mirrored from <https://github.com/luisalvarez96/Symmetries-in-Metabolic-Networks-of-E.-coli>. This includes a list of all complex building blocks found in the metabolic network of *E. coli*.

3. Codes and data for Chapters 17 and 18

Codes and data to find the symmetry-breaking circuits leading to various forms of flip-flops and oscillators used in Chapter 18 as well as motif finders developed by Leifer et al. (2020) are available at:

- Symmetry breaking Building blocks Finder Circuit— Algorithm for symmetry breaking circuits developed by Leifer et al. (2020) is available at <https://GitHub.com/MakseLab/CircuitFinder>.
- DDE solver— Delayed ODE (DDE) solver from Mathematica developed by Leifer et al. (2020) to study dynamics of building blocks at <https://GitHub.com/MakseLab/CircuitFinder/blob/master/delayODE.nb>.
- Data and analyzed networks for this chapter developed by Leifer et al. (2020) are available at <https://osf.io/jhmau>.

4. Codes and data for Chapter 19

This code enhances the fibration analysis by generating the minimal effective network, similar to a minimal genome or minimal TRN, for any given network.

- Codes and data to perform CoReSym, the complexity reduction analysis based on symmetries used in Chapter 19, and developed by Álvarez-García et al. (2025a), are available at <https://github.com/MakseLab/MinimalTRNCodes> mirrored from <https://github.com/luisalvarez96/MinimalTRN>. Further data is available at <https://osf.io/eh6ps>.

5. Codes and data for Chapter 20

These codes analyze the functional network of any system and produce the cluster synchronization of the network nodes. The analysis focuses on gene co-expression as a form of cluster synchronization.

- Correlations and synchronization measures and analysis. Code and gene co-expression data developed by Leifer et al. (2021) are available at <https://github.com/MakseLab/geneCoexpressionFibration>.

- Data, co-expression and building blocks of genetic networks developed by Leifer et al. (2021) are available at <https://osf.io/kbauj>.

6. Codes and data for Chapter 22

Perhaps the most important algorithms for practical applications are here. These codes deal with the problem of reconstructing an incomplete network guided by the observed cluster synchronization, as described in Section 22.2. The codes also solve the ‘one-to-many’ network inference problem of finding the activated pathways for a given synchrony pattern as explained in Section 22.2.1. These codes are then used in the brain analysis in Chapters 23, 24 and 25. We also include code and data to deal with similar problems: quasi-fibrations (Section 13.5.2) and pseudo-balanced colorings and network reconstructions using the symmetry-driven algorithm described in Section 13.5.3. This last algorithm is more general since it reconstructs a network without knowing the balanced coloring.

- The network reconstruction algorithm is based on mixed integer linear programming (MILP) and is guided by experimental cluster synchrony, as discussed in Section 22.5. This algorithm takes a balanced coloring and a baseline network to reconstruct or infer a network that satisfies the balanced coloring. This code addresses the inverse problem: given a functional network, it infers the corresponding structural network guided by cluster synchrony obtained experimentally. It was developed in (Ávila et al., 2025b; García-Hernández et al., 2025; Gili et al., 2025), and it is the basis for brain studies in Chapters 23, 24 and 25, respectively. It is available at https://github.com/MakseLab/genes_coloring.
- Pseudo-balanced coloring and network reconstruction via symmetries without knowing the balanced coloring. This integer programming developed by Leifer et al. (2022) solves a more difficult problem than in Section 22.5, since the coloring is not known *a priori*. It is available at <https://github.com/MakseLab/PseudoBalancedColoring>. An analogous problem is the link prediction code from (Leifer et al., 2022) available at <https://osf.io/26u3g>. Data and analyzed networks at <https://osf.io/prt5g>.
- Yet, another formulation for incomplete networks. Quasi-fibration finder and network reconstruction - Code for building quasi-fibration symmetries of graphs developed by Boldi et al. (2022) at <https://github.com/MakseLab/QuasiFibrations>. Data and analyzed networks at <https://osf.io/amswe>.

7. Codes and data for brain analysis for Chapters 23, 24, and 25

The main application of the inference and reconstruction algorithms from Chapter 22 is to infer brain networks. These algorithms are applied to three cases. More algorithms include functional brain network reconstructions and network analysis.

- *C. elegans* locomotion inference algorithm—Chapter 23. Code and data to reconstruct the connectome using synchronization data developed by Ávila et al. (2025b) are available at https://github.com/makselab/c_elegans_neural_network and <https://osf.io/9zvtq>.

- Minimal engram in the mouse brain— Chapter 24. Code and data for mouse connectome inference based on Allen connectome and engram data developed by García-Hernández et al. (2025) are available at <https://github.com/MakseLab/MouseConnectome>.
- Inferring the human language brain network— Chapter 25. Code and data to infer a human connectome developed by Gili et al. (2025) is available at <https://github.com/MakseLab>.
- Functional brain network construction from functional MRI signal— Series of codes and scripts perform a functional network inference construction from fMRI task-based data by Del Ferraro et al. (2018) at <https://github.com/gdelfe/Functional-network-construction-from-fMRI-signals>. Tools to construct functional networks and graph theoretical analysis at <http://kcorebrain.com>.
- Software to analyze brain images— Calculate influential areas, centralities, and k -core analysis developed by Li et al. (2020) is available at https://github.com/MakseLab/Healthy_brain_networks. A *C. elegans* locomotion simulator and network analysis developed by Ávila et al. (2025b) is available at https://github.com/MakseLab/C.-elegans_locomotive_simulator.

8. Codes and data for Chapter 26

- Code and data for the fibration analysis of deep neural networks developed by Velarde et al. (2025) are available at <https://osf.io/m4cxg>.

References

- Abrams, D. M., Pecora, L. M., and Motter, A. E. 2016. Introduction to focus issue: Patterns of network synchronization. *Chaos*, **26**(9), 094601.
- Agarwal, N., and Field, M J. 2010a. Decomposition of admissible functions in weighted coupled cell networks. *Nonlinearity*, **23**, 1269–1289.
- Agarwal, N., and Field, M J. 2010b. Dynamical equivalence of network architecture for coupled dynamical systems I: asymmetric inputs. *Nonlinearity*, **23**, 1245–1268.
- Aguiar, M., Bick, C., and Dias, A. 2023. Network dynamics with higher-order interactions: coupled cell hypernetworks for identical cells and synchrony. *Nonlinearity*, **36**, 4641–4673.
- Aguiar, M. A. D., and Dias, A. P. S. 2014. The lattice of synchrony subspaces of a coupled cell network: Characterization and computation algorithm. *J. Nonlinear Sci.*, **24**, 949–996.
- Aguiar, M. A. D., and Dias, A. P. S. 2018. Synchronization and equitable partitions in weighted networks. *Chaos*, **28**, 073105.
- Aguiar, M. A. D., and Dias, A. P. S. 2020. Synchrony and anti-synchrony in weighted networks. *SIAM J. Appl. Dyn. Sys.*, **20**, 1382–1420.
- Aguiar, M. A. D., Dias, A. P. S., and Ruan, H. 2022. Synchrony patterns in gene regulatory networks. *Physica D: Nonlinear Phenomena*, **429**, 133065.
- Ahnert, S. E., and Fink, T. M. A. 2016. Form and function in gene regulatory networks: the structure of network motifs determines fundamental properties of their dynamical state space. *J. R. Soc. Interface*, **13**(120), 20160179.
- Ahrens, M. B., Orger, M. B., Robson, D. N., Li, J. M., and Keller, P. J. 2013. Whole-brain functional imaging at cellular resolution using light-sheet microscopy. *Nature Methods*, **10**(5), 413–420.
- Ajo-Franklin, C. M., Drubin, D. A., Eskin, J. A., Gee, E. P., Landgraf, D., Phillips, I., and Silver, P. A. 2007. Rational design of memory in eukaryotic cells. *Genes Dev.*, **21**, 2271–2276.
- Albert, R., and Barabási, A.-L. 2002. Statistical mechanics of complex networks. *Rev. Mod. Phys.*, **74**(1), 47.
- Aldis, J. W. 2008. A polynomial time algorithm to determine maximal balanced equivalence relations. *Internat. J. Bif. Chaos*, **18**(02), 407–427.
- Aldis, J. W. 2010. *On Balance*. Ph.D. thesis, University of Warwick.
- Alivisatos, A P., Chun, M., Church, G. M., Deisseroth, K., Donoghue, J. P., Greenspan, R. J., McEuen, P. L., Roukes, M. L., Sejnowski, T. J., Weiss, P. S., et al. 2013. The brain activity map. *Science*, **339**(6125), 1284–1285.

- Alon, U. 2019. *An Introduction to Systems Biology: Design Principles of Biological Circuits*. CRC Press.
- Álvarez-García, L. A., Liebermeister, W., Leifer, I., and Makse, H. A. 2025a. *Complexity reduction by symmetry: uncovering the minimal regulatory network for logical computation in bacteria*. preprint arXiv:2310.10895, PLoS Comp. Biol.
- Álvarez-García, L. A., Liu, H., Ishida, C., Sánchez-Pérez, M., Wuchty, S., and Makse, H. A. 2025b. *Symmetries in metabolic networks of E. coli*. preprint PNAS Nexus.
- Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., et al. 2021. Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *J. Big Data*, **8**(1), 53.
- Anderson, J. C., Voigt, C. A., and Arkin, A. P. 2007. Environmental signal integration by a modular AND gate. *Mol. Syst. Biol.*, **3**, 133.
- Anderson, P. W. 1972. More is different: broken symmetry and the nature of the hierarchical structure of science. *Science*, **177**(4047), 393–396.
- Angluin, D. 1980. Local and global properties in networks of processors. Pages 82–93 of: *Proceedings of the Twelfth Annual ACM Symposium on Theory of Computing*.
- Arenas, A., Díaz-Guilera, A., Kurths, J., Moreno, Y., and Zhou, C. 2008. Synchronization in complex networks. *Phys. Rep.*, **469**(3), 93–153.
- Arese-Lucini, F., Morone, F., Tomassone, M. S., and Makse, H. A. 2020. Diversity increases the stability of ecosystems. *PLoS ONE*, **15**(4), e0228692.
- Arnold, V. I. 1989. *Mathematical Methods of Classical Mechanics*. Springer.
- Atkinson, D. E. 1970. A dualism in biology: symmetry and function of biological systems at the macromolecular level. Proceedings of the 11th Nobel Symposium, Södergarn, Lidingö, Sweden, Aug. 1968. A. Engström and B. Strandberg, Eds. Interscience (Wiley), New York, and Almqvist and Wiksell, Stockholm. *Science*, **168**(3932), 723–724.
- Atkinson, M. R., Savageau, M. A., Myers, J. T., and Ninfa, A. J. 2003. Development of genetic circuitry exhibiting toggle switch or oscillatory behavior in *Escherichia coli*. *Cell*, **113**, 597–607.
- Ávila, B., Serafino, M., Augusto, P., Zimmer, M., and Makse, H. A. 2024. Fibration symmetries and cluster synchronization in the *Caenorhabditis elegans* connectome. *PLoS ONE*, **19**, e0297669.
- Ávila, B., Phillips, D., Gili, T., Serafino, M., Zimmer, M., and Makse, H. A. 2025a. *Link prediction in biological networks guided by symmetry-driven synchronization dynamics*. preprint.
- Ávila, B., Augusto, P., Hashimi, A., Phillips, D., Gili, T., Zimmer, M., and Makse, H. A. 2025b. *Symmetries and synchronization from whole-neural activity in C. elegans connectome: Integration of functional and structural networks*. preprint arXiv:2409.02682.
- Babai, L. 1996. Automorphism groups, isomorphism, reconstruction. Pages 1447–1540 of: *Handbook of Combinatorics* (vol. 2). MIT Press.
- Babai, L. 2016. Graph isomorphism in quasipolynomial time [Extended Abstract]. Pages 684–697 of: *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*. STOC '16.
- Baker, B., Lansdell, B., and Kording, K. P. 2022. Three aspects of representation in neuroscience. *Trends Cognitive Sci.*, **26**(11), 942–958.

- Balázs, G., Heath, A. P., Shi, L., and Gennaro, M. L. 2008. The temporal response of the *Mycobacterium tuberculosis* gene regulatory network during growth arrest. *Mol. Syst. Biol.*, **4**, 225.
- Bansal, M., Belcastro, V., Ambesi-Impiombato, A., and Di Bernardo, D. 2007. How to infer gene networks from expression profiles. *Mol. Syst. Biol.*, **3**(1), 78.
- Bao, J., He, Y., Hirst, E., and Pietromonaco, S. 2020. Lectures on the Calabi-Yau landscape. *arXiv*, 2001.01212.
- Barabási, A.-L. 2009. Scale-free networks: a decade and beyond. *Science*, **325**(5939), 412–413.
- Barabási, A.-L., and Albert, R. 1999. Emergence of scaling in random networks. *Science*, **286**(5439), 509–512.
- Barabási, A.-L., and Pósfai, M. 2016. *Network Science*. Cambridge University Press.
- Barahona, M., and Pecora, L. M. 2002. Synchronization in small-world systems. *Phys. Rev. Lett.*, **89**(5), 054101.
- Barbuti, R., Gori, R., Milazzo, P., and Nasti, L. 2020. A survey of gene regulatory networks modelling methods: from differential equations, to Boolean and qualitative bioinspired models. *J. Membrane Comput.*, **2**(3), 207–226.
- Bardella, G., Bifone, A., Gabrielli, A., Gozzi, A., and Squartini, T. 2016. Hierarchical organization of functional connectivity in the mouse brain: a complex network approach. *Sci. Rep.*, **6**, 32060.
- Bargmann, C. I. 1993. Genetic and cellular analysis of behavior in *C. elegans*. *Annu. Rev. Neurosci.*, **16**, 47–61.
- Barrett, T., Wilhite, S. E., Ledoux, P., Evangelista, C., Kim, I. F., Tomashevsky, M., Marshall, K. A., Phillip, K. H., Sherman, P. M., Holko, M., et al. 2012. NCBI GEO: archive for functional genomics data sets—update. *Nucleic Acids Res.*, **41**(D1), D991–D995.
- Bassett, D. S., and Bullmore, E. D. 2006. Small-world brain networks. *The Neuroscientist*, **12**(6), 512–523.
- Basu, S., Mehreja, R., Thibierge, S., Chen, M. T., and Weiss, R. 2004. Spatiotemporal control of gene expression with pulse-generating networks. *Proc. Natl. Acad. Sci. USA.*, **101**, 6355–6360.
- Battiston, Federico, Cencetti, Giulia, Iacopini, Iacopo, Latora, Vito, Lucas, Maxime, Pata-nia, Alice, Young, Jean-Gabriel, and Petri, Giovanni. 2020. Networks beyond pairwise interactions: structure and dynamics. *Phys. Rep.*, **874**, 1–92.
- Belykh, I., and Hasler, M. 2011. Mesoscale and clusters of synchrony in networks of bursting neurons. *Chaos*, **21**(1), 016106.
- Belykh, I., and Hasler, M. 2015. Patterns of synchrony in neuronal networks: the role of synaptic inputs. Pages 1–28 of: *Nonlinear Dynamics New Directions*. Springer.
- Belykh, V. N., Osipov, G. V., Petrov, V. S., Suykens, J. A. K., and Vandewalle, J. 2008. Cluster synchronization in oscillatory networks. *Chaos*, **18**(3), 037106.
- Berge, C. 1973. *Graphs and Hypergraphs*. North Holland.
- Berge, C. 1984. *Hypergraphs: Combinatorics of Finite Sets*. Vol. 45. Elsevier.
- Berge, C. 2001. *The Theory of Graphs*. Courier Corporation.

- Berghofer, P., François, J., Friederich, S., Gomes, H., Hetzroni, G., Maas, A., and Sonnenheimer, R. 2023. Gauge Symmetries, Symmetry Breaking, and Gauge-Invariant Approaches. *Elements in the Foundations of Contemporary Physics*, July.
- Berkholz, C., Bonsma, P., and Grohe, M. 2017. Tight lower and upper bounds for the complexity of canonical colour refinement. *Theory Comput. Syst.*, **60**, 581–614.
- Berman, G. J., Choi, D. M., Bialek, W., and Shaevitz, J. W. 2014. Mapping the stereotyped behaviour of freely moving fruit flies. *J. Roy. Soc. Interface*, **11**(99), 20140672.
- Bernal, J. D. 1967. Symmetry of the genesis of form. *J. Mol. Biol.*, **24**(3), 379–390.
- Bertsimas, D., and Tsitsiklis, J. N. 1997. *Introduction to Linear Optimization*. Vol. 6. Athena Scientific. Belmont, MA.
- Bhati, N. P., and Szegö, G. P. 1970. *Stability Theory of Dynamical Systems*. Springer.
- Bintu, L., Buchler, N. E., García, H. G., Gerland, U., Hwa, T., Kondev, J., and Phillips, R. 2005. Transcriptional regulation by the numbers: models. *Curr. Opinion Genet. Develop.*, **15**(2), 116–124.
- Blaha, K. A., Huang, K., Della Rossa, F., Pecora, L., Hossein-Zadeh, M., and Sorrentino, F. 2019. Cluster synchronization in multilayer networks: A fully analog experiment with LC oscillators with physically dissimilar coupling. *Phys. Rev. Lett.*, **122**(1), 014101.
- Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. 2008. Fast unfolding of communities in large networks. *J. Stat. Mech.: Theo. & Expt.*, **2008**(10), P10008.
- Boccaletti, S., Pecora, L. M., and Pelaez, A. 2001. A unifying framework for synchronization of coupled dynamical systems. *Phys. Rev. E.*, **63**, 066219.
- Boldi, P. 2023. The Emergence of Hypergraphs in Complex System Analysis. Pages 1–7 of: *2023 IEEE 9th International Conference on Collaboration and Internet Computing (CIC)*. IEEE.
- Boldi, P., and Stewart, I. 2025. Branching ratios of input trees for directed multigraphs. *arXiv*, 2501.06812.
- Boldi, P., and Vigna, S. 1997. Self-stabilizing universal algorithms. Pages 141–156 of: WSS. Citeseer.
- Boldi, P., and Vigna, S. 2001. An effective characterization of computability in anonymous networks. Pages 33–47 of: *International Symposium on Distributed Computing*. Springer.
- Boldi, P., and Vigna, S. 2002a. Fibrations of graphs. *Discr. Math.*, **243**(1-3), 21–66.
- Boldi, P., and Vigna, S. 2002b. Universal dynamic synchronous self-stabilization. *Distributed Computing*, **15**(3), 137–153.
- Boldi, P., Lonati, V., Santini, M., and Vigna, S. 2006. Graph fibrations, graph isomorphism, and PageRank. *RAIRO-Theoret. Informatics & Appl.-Informat. Théor. & Appl.*, **40**(2), 227–253.
- Boldi, P., Leifer, I., and Makse, H. A. 2022. Quasi-fibrations of graphs to find symmetries in biological networks. *J. Stat. Mech.: Theo. and Exp.*, **2022** (11), 113401.
- Bollen, J., Steegmans, J., Bussche, J. Van den, and Vansumeren, S. 2023 (June). Learning Graph Neural Networks using Exact Compression. Pages 1–9 of: *Proceedings of the 6th Joint Workshop on Graph Data Management Experiences & Systems (GRADES) and Network Data Analytics (NDA)*. arXiv:2304.14793 [cs].
- Bollobás, B. 2012. *Graph Theory: An introductory course*. Vol. 63. Springer Science & Business Media.

- Bourbaki, N. 1994. *Elements of the History of Mathematics*. Springer-Verlag, Berlin.
- Brandt, H. 1927. Über eine verallgemeinerung des gruppenbegriffes. *Math. Annu.*, **96**(1), 360–366.
- Brenner, S. 1974. The genetics of *Caenorhabditis elegans*. *Genetics*, **77**, 71–94.
- Bronstein, M. M., Bruna, J., Cohen, T., and Veličković, P. 2021. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv*, 2104.13478.
- Brown, R. 2006. *Topology and Groupoids*. Groupoids.org.uk, Deganwy.
- Brugere, I., Gallagher, B., and Berger-Wolf, T. Y. 2018. Network structure inference, a survey: Motivations, methods, and applications. *ACM Comput. Surv. (CSUR)*, **51**(2), 1–39.
- Bruña, R., Maestú, F., and Pereda, E. 2018. Phase locking value revisited: teaching new tricks to an old dog. *J. Neural Eng.*, **15**(5), 056011.
- Buchanan, M., Caldarelli, G., De Los Rios, P., Rao, F., and Vendruscolo, M. 2010. *Networks in Cell Biology*. Cambridge University Press.
- Buchler, N. E., Gerland, U., and Hwa, T. 2003. On schemes of combinatorial transcription logic. *Proc. Natl. Acad. Sci. USA*, **100**(9), 5136–5141.
- Bullmore, E., and Sporns, O. 2009. Complex brain networks: graph theoretical analysis of structural and functional systems. *Nat. Rev. Neurosci.*, **10**(3), 186–198.
- Bunde, A., and Havlin, S. 1999. *Fractals and Disordered Systems, 2nd edition*. Springer-Verlag, Heidelberg.
- Butte, A. J., and Kohane, I. S. 1999. Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements. Pages 418–429 of: *Biocomputing 2000*. World Scientific.
- Cahn, R. N. 1996. The eighteen arbitrary parameters of the standard model in your everyday life. *Rev. Mod. Phys.*, **68**(3), 951.
- Cameron, D. E., Bashor, C. J., and Collins, J. J. 2014. A brief history of synthetic biology. *Nat. Rev. Microbiol.*, **12**(5), 381–390.
- Cameron, P. J. 2006. Graph homomorphisms. *Combinatorics Study Group Notes*, 1–7.
- Cardon, A., and Crochemore, M. 1982. Partitioning a graph in O (—A—log2—V—). *Theor. Comp. Sci.*, **19**(1), 85–98.
- Chalfie, M., Sulston, J. E., White, J. G., Southgate, E., Thomson, J. N., and Brenner, S. 1985. The neural circuit for touch sensitivity in *Caenorhabditis elegans*. *J. Neurosci.*, **5**(4), 956–964.
- Chang, W., Wang, L., Yang, R., Wang, X., Gao, Z., and Zhou, X. 2023. Representing linguistic communicative functions in the premotor cortex. *Cerebral Cortex*, **33**(9), 5671–5689.
- Chaplain, M. A. J., Singh, G. D., and McLachlan, J. C. (eds). 1999. *On Growth and Form: Spatio-temporal Pattern Formation in Biology*. Wiley.
- Chen, B. L., Hall, D. H., and Chklovskii, D. B. 2006. Wiring optimization can relate neuronal structure and function. *Proc. Natl. Acad. Sci. USA*, **103**(12), 4723–4728.
- Chen, H., Li, X., and Huang, Z. 2005. Link prediction approach to collaborative filtering. Pages 141–142 of: *Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'05)*. IEEE.

- Chen, Y., Zhu, J., Lum, P. Y., Yang, X., Pinto, S., MacNeil, D. J., Zhang, C., Lamb, J., Edwards, S., Sieberts, S., et al. 2008. Variations in DNA elucidate molecular networks that cause disease. *Nature*, **452**(7186), 429–435.
- Cho, A. Nov 10, 2015. *Mathematician claims breakthrough in complexity theory*. <https://www.science.org/content/article/mathematician-claims-breakthrough-complexity-theory> [Accessed Jul 2022].
- Choi, P. J., Cai, L., Frieda, F., and Sunney Xie, X. 2008. A stochastic single-molecule event triggers phenotype switching of a bacterial cell. *Science*, **322**(5900), 442–446.
- Clauset, A., Moore, C., and Newman, M. E. J. 2008. Hierarchical structure and the prediction of missing links in networks. *Nature*, **453**(7191), 98–101.
- Cohen, R. L. 1998. The Topology of Fiber Bundles. *Lecture Notes, Stanford University*.
- Consortium, The UniProt. 2022. UniProt: the Universal Protein Knowledgebase in 2023. *Nucleic Acids Res.*, **51**(D1), D523–D531.
- Conte, D., Foggia, P., Sansone, C., and Vento, M. 2004. Thirty years of graph matching in pattern recognition. *Internat. J. Pattern Recog. & Artificial Intelligence*, **18**(03), 265–298.
- Cook, S. A. 1971. The complexity of theorem-proving procedures. Pages 151–158 of: *Proceedings of the Third Annual ACM Symposium on Theory of Computing*.
- Coram, D. S., and Duvall, P. F. 1977. Approximate fibrations. *Rocky Mountain J. Math.*, **7**(2), 275–288.
- Cordella, L. P., Foggia, P., Sansone, C., and Vento, M. 2004. A (sub) graph isomorphism algorithm for matching large graphs. *IEEE Trans. Pattern Analysis & Machine Intelligence*, **26**(10), 1367–1372.
- Corneil, D. G., and Gotlieb, C. C. 1970. An efficient algorithm for graph isomorphism. *J. ACM*, **17**(1), 51–64.
- Creamer, M. S., Chen, K. S., Leifer, A. M., and Pillow, J. W. 2022. Correcting motion induced fluorescence artifacts in two-channel neural imaging. *PLoS Comput. Biol.*, **18**(9), e1010421.
- Creutz, M. 1983. *Quarks, Gluons and Lattices*. Vol. 8. Cambridge University Press.
- Crick, F. 1970. Central dogma of molecular biology. *Nature*, **227**(5258), 561–563.
- Crick, F. 1988. *What Mad Pursuit: A Personal View of Scientific Discovery*. Basic Books, New York.
- Crooks, G. E., Hon, G., Chandonia, J.-M., and Brenner, S. E. 2004. WebLogo: a sequence logo generator. *Genome research*, **14**(6), 1188–1190.
- Csárdi, G., and Nepusz, T. 2006. *The igraph software package for complex network research*.
- Cvetković, D. M., Doob, M., and Sachs, H. 1998. *Spectra of Graphs: Theory and Applications*, 3rd rev. enl. ed. New York: Wiley.
- Dahms, T., Lehnert, J., and Schöll, E. 2012. Cluster and group synchronization in delay-coupled networks. *Phys. Rev. E*, **86**(1), 016202.
- Dalchau, N., Szép, G., Hernansaiz-Ballesteros, R., Barnes, C. P., Cardelli, L., Phillips, A., and Csikász-Nagy, A. 2018. Computing with biological switches and clocks. *Natural Computing*, **17**(4), 761–779.

- Darga, P. T., Sakallah, K. A., and Markov, I. L. 2008. Faster Symmetry Discovery Using Sparsity of Symmetries. Pages 149–154 of: *Proceedings of the 45th Annual Design Automation Conference*.
- Davey, B. A., and Priestley, H. A. 2010. *Introduction to Lattices and Order*. Cambridge University Press.
- De Nardo, L. A., Liu, C. D., Allen, W. E., Adams, E. L., Friedmann, D., Dadgar-Kiani, E., Fu, L., Guenthner, C. J., Lee, J. H., Tessier-Lavigne, M., and Luo, L. 2019. Temporal evolution of cortical ensembles promoting remote memory retrieval. *Nat. Neurosci.*, **22**, 460–469.
- Deco, G., Ponce-Alvarez, A., Mantini, D., Romani, G. L., Hagmann, P., and Corbetta, M. 2013. Resting-state functional connectivity emerges from structurally and dynamically shaped slow linear fluctuations. *J. Neurosci.*, **33**(27), 11239–11252.
- Del Ferraro, G., Moreno, A., Min, B., Morone, F., Pérez-Ramírez, U., Pérez-Cervera, L., Parra, L. C., Holodny, A., Canals, S., and Makse, H. A. 2018. Finding influential nodes for integration in brain networks using optimal percolation theory. *Nat. Commun.*, **9**(1), 2274.
- Della Rossa, F., Pecora, L., Blaha, K., Shirin, A., Klickstein, I., and Sorrentino, F. 2020. Symmetries and cluster synchronization in multilayer networks. *Nature Commun.*, **11**(1), 3179.
- Deng, C., Zhao, Z., Wang, Y., Zhang, Z., and Feng, Z. 2020 (Feb.). *GraphZoom: A multi-level spectral approach for accurate and scalable graph embedding*. arXiv:1910.02370 [cs, stat].
- DeVille, L., and Lerman, E. 2015a. Dynamics on networks of manifolds. *SIGMA. Symmetry, Integrability and Geometry: Methods and Applications*, **11**, 022.
- DeVille, L., and Lerman, E. 2015b. Modular dynamical systems on networks. *J. Eur. Math. Soc.*, **17**, 2977–3013.
- Dheeraj, S., Arons, A., Mitchell, T. I., Pignatelli, M., Ryan, T. J., and Tonegawa, S. 2016. Memory retrieval by activating engram cells in mouse models of early Alzheimer’s disease. *Nature*, **531**, 508–512.
- Díaz-Parra, A., Pérez-Ramírez, U., Pacheco-Torres, J., Pfarr, S., Sommer, W. H., Moratal, D., and Canals, S. 2017. Evaluating network brain connectivity in alcohol postdependent state using Network-Based Statistic. Pages 533–536 of: *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE.
- Dixon, J. D., and Mortimer, B. 1996. *Permutation Groups*. Vol. 163. Springer Science & Business Media.
- Do, A.-L., Höfener, J., and Gross, T. 2012. Engineering mesoscale structures with distinct dynamical implications. *New J. Phys.*, **14**(11), 115022.
- do Carmo, M. P. 1992. *Riemannian Geometry*. Birkhäuser.
- Dobrin, R., Beg, Q. K., Barabási, A.-L., and Oltvai, Z. N. 2004. Aggregation of topological motifs in the *{Escherichia} coli* transcriptional regulatory network. *BMC Bioinformatics*, **5**(1), 1–8.
- Driver, R. D. 2012. *Ordinary and Delay Differential Equations*. Vol. 20. Springer Science & Business Media.

- Economou, A. D., Atsushi, O., Thantrira, P., Sharpe, P. T., Kondo, S., Basson, M. A., Gritli-Linde, A., Cobourne, M. T., and Green, J. B. A. 2012. Periodic stripe formation by a Turing mechanism operating at growth zones in the mammalian palate. *Nature Genetics*, **44**, 348–351.
- Eilenberg, S., and MacLane, S. 1945. General theory of natural equivalences. *Trans. Amer. Math. Soc.*, **58**(2), 231–294.
- Eisen, M. B., Spellman, P. T., Brown, P. O., and Botstein, D. 1998. Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci. USA*, **95**(25), 14863–14868.
- Elowitz, M. B., and Leibler, S. 2000. A synthetic oscillatory network of transcriptional regulators. *Nature*, **403**(6767), 335–338.
- Field, M. 2004. Combinatorial dynamics. *Dynamical Systems*, **19**, 217–243.
- FitzHugh, R. 1961. Impulses and physiological states in theoretical models of nerve membrane. *Biophys. J.*, **1**(6), 445–466.
- Fodor, J., Brand, M., Stones, R. J., and Buckle, A. M. 2020. Intrinsic limitations in mainstream methods of identifying network motifs in biology. *BMC Bioinformatics*, **6:9121:165**, doi.org/10.1186/s12859-020-3441-x.
- Fontana, W., and Schuster, P. 1998. Continuity in evolution: on the nature of transitions. *Science*, **280**(5368), 1451–1455.
- Fortunato, S. 2010. Community detection in graphs. *Phys. Rep.*, **486**, 75–174.
- Friedman, J., Hastie, T., and Tibshirani, R. 2008. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics*, **9**(3), 432–441.
- Friston, K. J. 2011. Functional and effective connectivity: a review. *Brain connectivity*, **1**(1), 13–36.
- Frucht, R. 1949. Graphs of degree three with a given abstract group. *Canad. J. Math.*, **1**, 365–378.
- Fu, C., Lin, W., Huang, L., and Wang, X. 2014. Synchronization transition in networked chaotic oscillators: The viewpoint from partial synchronization. *Phys. Rev. E*, **89**(5), 052908.
- Fung, E., Wong, W. W., Suen, J. K., Bulter, T., Lee, S. G., and Liao, J. C. 2005. A synthetic gene-metabolic oscillator. *Nature*, **435**, 118–122.
- Fussenegger, M. 2010. Synchronized bacterial clocks. *Nature*, **463**(7279), 301–302.
- Gallos, L. K., Sigman, M., and Makse, H. A. 2012a. The conundrum of functional brain networks: small-world efficiency or fractal modularity. *Front. Physiol.*, **3**, 123.
- Gallos, L. K., Makse, H. A., and Sigman, M. 2012b. A small world of weak ties provides optimal global integration of self-similar modules in functional brain networks. *Proc. Natl. Acad. Sci. USA*, **109**(8), 2825–2830.
- Gama-Castro, S., Rinaldi, F., López-Fuentes, A., Balderas-Martínez, Y. I., Clematide, S., Ellendorff, T. R., Santos-Zavaleta, A., Marques-Madeira, H., and Collado-Vides, J. 2014. Assisted curation of regulatory interactions and growth conditions of OxyR in *E. coli* K-12. *Database*, **2014**, bau049.
- Gama-Castro, S., Salgado, H., Santos-Zavaleta, A., Ledezma-Tejeida, D., Muñiz-Rascado, L., García-Sotelo, J. S., Alquicira-Hernández, K., Martínez-Flores, I., Pannier, L., Castro-Mondragón, J. A., et al. 2016. RegulonDB version 9.0: high-level integration of gene

- regulation, coexpression, motif clustering and beyond. *Nucleic Acids Res.*, **44**(D1), D133–D143.
- García-Hernández, R., Álvarez-García, L., Gili, T., Canals, S., and Makse, H. A. 2025. *Local symmetries connect the structure and function of memory engrams in the mouse brain*. preprint.
- Gardner, T. S., Cantor, C. R., and Collins, J. J. 2000. Construction of a genetic toggle switch in *Escherichia coli*. *Nature*, **403**(6767), 339–342.
- Garey, M. R., and Johnson, D. S. 1979. *Computers and Intractability*. Vol. 174. Freeman.
- Gell-Mann, M. 1995. *The Quark and the Jaguar: Adventures in the Simple and the Complex*. Macmillan.
- Georgi, H. 2018. *Lie Algebras in Particle Physics: From isospin to unified theories*. CRC Press.
- Getoor, L., and Diehl, C. P. 2005. Link mining: a survey. *Acm Sigkdd Explorations Newsletter*, **7**(2), 3–12.
- Gili, T., Avila, B., Pasquini, L., Holodny, A., Phillips, D., Boldi, P., Gabrielli, A., Caldarelli, G., Zimmer, M., and Makse, H. A. 2025. *Fibration symmetry-breaking supports functional transitions in a brain network engaged in language*. preprint.
- Girvan, M., and Newman, M. E. J. 2002. Community structure in social and biological networks. *Proc. Natl. Acad. Sci. USA*, **99**(12), 7821–7826.
- Glashow, S. L. 1961. Partial-symmetries of weak interactions. *Nuclear Phys.*, **22**, 579–588.
- Glass, L., and Kauffman, S. A. 1973. The logical analysis of continuous, non-linear biochemical control networks. *J. Theoret. Biol.*, **38**, 103–129.
- Golubitsky, M., and Stewart, I. 1985. Hopf bifurcation with dihedral group symmetry: coupled nonlinear oscillators. Pages 131–173 of: Golubitsky, M., and Guckenheimer, J. (eds), *Proceedings of the AMS-IMS-SIAM Joint Summer Research Conference, July 1985, Arcata*. Amer. Math. Soc.
- Golubitsky, M., and Stewart, I. 2003. *The Symmetry Perspective: From equilibrium to chaos in phase space and physical space*. Vol. 200. Springer Science & Business Media.
- Golubitsky, M., and Stewart, I. 2006. Nonlinear dynamics of networks: the groupoid formalism. *Bull. Amer. Math. Soc.*, **43**(3), 305–364.
- Golubitsky, M., and Stewart, I. 2015. Symmetry methods in mathematical biology. *São Paulo J. Math. Sci.*, **9**(1), 1–36.
- Golubitsky, M., and Stewart, I. 2016. Rigid patterns of synchrony for equilibria and periodic cycles in network dynamics. *Chaos*, **26**(9), 094803.
- Golubitsky, M., and Stewart, I. 2023. *Dynamics and Bifurcation in Networks*. SIAM.
- Golubitsky, M., Stewart, I., and Schaeffer, David. G. 1988. *Singularities and Groups in Bifurcation Theory: Volume II*. Vol. 69. Springer Science & Business Media.
- Golubitsky, M., Stewart, I., and Török, A. 2005a. Patterns of synchrony in coupled cell networks with multiple arrows. *SIAM J. Appl. Dyn. Sys.*, **4**(1), 78–100.
- Golubitsky, M., Stewart, I., and Török, A. 2005b. Patterns of synchrony in coupled cell networks with multiple arrows. *SIAM J. Appl. Dyn. Sys.*, **4**(1), 78–100.
- Goodwin, B. C. 1963. *Temporal Organization in Cells: A dynamic theory of cellular control processes*. Academic Press.

- Gray, J. M., Hill, J. J., and Bargmann, C. I. 2005. A circuit for navigation in *Caenorhabditis elegans*. *Proc. Natl. Acad. Sci. USA*, **102**(9), 3184–3191.
- Greicius, M. D., Supekar, K., Menon, V., and Dougherty, R. F. 2009. Resting-state functional connectivity reflects structural connectivity in the default mode network. *Cereb. Cortex*, **19**, 72–78.
- Greiner, W., Schramm, S., and Stein, E. 2007. *Quantum Chromodynamics*. Springer.
- Grohe, M., and Schweitzer, P. 2020. The Graph Isomorphism Problem. *Commun. ACM*, **63**(11), 128–134.
- Gromov, M. 2011. Crystals, proteins, stability and isoperimetry. *Bull. Amer. Math. Soc.*, **48**(2), 229–257.
- Grothendieck, A. 1959. Technique de descente et théorèmes d’existence en géométrie algébrique. I. Généralités. Descente par morphismes fidèlement plats. *Séminaire Bourbaki*, **5**, 299–327.
- Grove, C. L., and Gunsalus, R. P. 1987. Regulation of the aroH operon of *Escherichia coli* by the tryptophan repressor. *J. Bacteriol.*, **169**(5), 2158–2164.
- Gruber, H. 2012. Digraph complexity measures and applications in formal language theory. *Discrete Math. & Theoret. Comp. Sci.*, **14**.
- Guckenheimer, J., and Holmes, P. 1983. *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields*. Springer.
- Guet, C. C., Elowitz, M. B., Hsing, W., and Leibler, S. 2002. Combinatorial synthesis of genetic networks. *Science*, **296**, 1466–1470.
- Gurobi Optimization, LLC. 2023. *Gurobi Optimizer Reference Manual*.
- Hagmann, P., Cammoun, L., Gigandet, X., Meuli, R., Honey, C. J., Wedeen, V. J., and Sporns, O. 2008. Mapping the structural core of human cerebral cortex. *PLoS Biol.*, **6**(7), e159.
- Hahn, G., and Tardif, C. 1997. Graph homomorphisms: structure and symmetry. Pages 107–166 of: *Graph Symmetry*. Springer.
- Hall, D. H., and Russell, R. L. 1991. The posterior nervous system of the nematode *Caenorhabditis elegans*: serial reconstruction of identified neurons and complete pattern of synaptic interactions. *J. Neurosci.*, **11**(1), 1–22.
- Hallinen, K. M., Dempsey, R., Scholz, M., Yu, X., Linder, A., Randi, F., Sharma, A. K., Shaevitz, J. W., and Leifer, A. M. 2021. Decoding locomotion from population neural activity in moving *C. elegans*. *Elife*, **10**, e66135.
- Ham, T. S., Lee, S. K., Keasling, J. D., and Arkin, A. P. 2008. Design and construction of a double inversion recombination switch for heritable sequential genetic memory. *PLoS One*, **3**, e2815.
- Hamermesh, M. 2012. *Group Theory and Its Application to Physical Problems*. Courier Corporation.
- Hammack, R. H., Imrich, W., Klavžar, S., and Klavžar, S. 2011. *Handbook of Product Graphs*. Vol. 2. CRC press Boca Raton.
- Hammoud, Z., and Kramer, F. 2020. Multilayer networks: aspects, implementations, and application in biomedicine. *Big Data Anal.*, **5**(1), 2.

- Han, H., Cho, J.-W., Lee, S., Yun, A., Kim, H., Bae, D., Yang, S., Yeong Kim, C., Lee, M., Kim, E., Lee, S., Kang, B., Jeong, D., Kim, Y., Jeon, H.-N., Jung, H., Nam, S., Chung, M., Kim, J.-H., and Lee, I. 2017. TRRUST v2: An expanded reference database of human and mouse transcriptional regulatory interactions. *Nucleic Acids Res.*, **46**, D380–D386.
- Harary, F. 1969. *Graph Theory*. Addison-Wesley.
- Harary, F. 1993. *Graph Theory*. Addison-Wesley.
- Hartwell, L. H., Hopfield, J. J., Leibler, S., and Murray, A. W. 1999a. From molecular to modular cell biology. *Nature*, **402**(6761), C47–C52.
- Hartwell, L. H., Hopfield, J. J., Leibler, S., and Murray, A. W. 1999b. From molecular to modular cell biology. *Nature*, **402**(Suppl 6761), C47–C52.
- Hasan, A. B. M., Kurata, H., and Pechmann, S. 2019. Improvement of the memory function of a mutual repression network in a stochastic environment by negative autoregulation. *BMC Bioinformatics*, **20**(1), 1–14.
- Hassard, B. D., Kazarinoff, N. D., and Wan, Y.-H. 1981. *Theory and Applications of Hopf Bifurcation*. Vol. 41. Cambridge University Press.
- Hasty, J., Dolnik, M., Rottschäfer, V., and Collins, J. J. 2002. Synthetic gene network for entraining and amplifying cellular oscillations. *Phys. Rev. Lett.*, **88**(14), 148101.
- Hebb, D. O. 1949. *The Organization of Behavior*. Wiley.
- Hell, P., and Nesetril, J. 2004. *Graphs and Homomorphisms*. Oxford University Press.
- Hertrich, I., Dietrich, S., and Ackermann, H. 2016. The role of the supplementary motor area for speech and language processing. *Neurosci. & Biobehavioral Rev.*, **68**, 602–610.
- Hertz, G. Z., Hartzell III, G. W., and Stormo, G. D. 1990. Identification of consensus patterns in unaligned DNA sequences known to be functionally related. *Bioinformatics*, **6**(2), 81–92.
- Hickok, G. 2022. The dual stream model of speech and language processing. *Handbook of Clinical Neurology*, **185**, 57–69.
- Hickok, G., and Poeppel, D. 2007. The cortical organization of speech processing. *Nat. Rev. Neurosci.*, **8**(5), 393–402.
- Higgins, I., Racanière, S., and Rezende, D. 2022. Symmetry-Based Representations for Artificial and Biological General Intelligence. *Front. Comput. Neurosci.*, **16**.
- Higgins, P. J. 1971. *Notes on Categories and Groupoids*. Van Nostrand Reinhold.
- Hirsch, M. W., and Smale, S. 1974. *Differential Equations, Dynamical Systems, and Linear Algebra*. Academic Press.
- Hodgkin, A. L., and Huxley, A. F. 1952. A quantitative description of membrane current and its application to conduction and excitation in nerve. *J. Physiol.*, **117**(4), 500–544.
- Holodny, A. I., Schulder, M., Ybasco, A., and Liu, W. C. 2002. Translocation of Broca's area to the contralateral hemisphere as the result of the growth of a left inferior frontal glioma. *J. Comput. Assisted Tomography*, **26**(6), 941–943.
- Honey, C. J., Sporns, O., Cammoun, L., Gigandet, X., Thiran, J.-P., Meuli, R., and Hagmann, P. 2009. Predicting human resting-state functional connectivity from structural connectivity. *Proc. Natl. Acad. Sci. USA*, **106**(6), 2035–2040.
- Hopcroft, J. 1971. An $n \log n$ algorithm for minimizing states in a finite automaton. Pages 189–196 of: *Theory of Machines and Computations*. Elsevier.

- Horowitz, P., and Hill, W. 2015. *The Arts of Electronics*. 3rd ed. Cambridge University Press, New York.
- Horvath, S. 2011. *Weighted Network Analysis: Applications in genomics and systems biology*. Springer Science & Business Media.
- Huang, L., Chen, Q., Lai, Y.-C., and Pecora, L. 2009. Generic behaviour of master-stability functions in coupled nonlinear dynamical systems. *Phys. Rev. E*, **80**, 036204.
- Huang, N. T., and Villar, S. 2021. A short tutorial on the weisfeiler-lehman test and its variants. Pages 8533–8537 of: *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE.
- Husemoller, D. 1966. *Fibre Bundles*. Vol. 5. Springer.
- Ibort, A., and Rodriguez, M.A. 2020. *An Introduction to Groups, Groupoids, and their Representations*. CRC Press.
- Ilinski, K. 2001. *Physics of Finance: Gauge modelling in non-equilibrium pricing*. Wiley.
- Imrich, W., and Izbicki, H. 1975. Associative products of graphs. *Monatshefte für Math.*, **80**(4), 277–281.
- Ingram, P. J., Stumpf, M. P. H., and Stark, J. 2006. Network motifs: structure does not determine function. *BMC Genomics*, **7**(1), 1–12.
- Izhikevich, E. M. 2003. Simple model of spiking neurons. *IEEE Trans. Neural Net.*, **14**(6), 1569–1572.
- Jackson, J. D. 2012. *Classical Electrodynamics*. Wiley.
- Jacob, F. 1977. Evolution and tinkering. *Science*, **196**(4295), 1161–1166.
- Jacob, F., and Monod, J. 1961. Genetic regulatory mechanisms in the synthesis of proteins. *J. Mol. Biol.*, **3**(3), 318–356.
- Jänicke, St., Heine, C., Hellmuth, M., Stadler, P. F., and Scheuermann, G. 2010. Visualization of graph products. *IEEE Trans. Visualization Comp. Graphics*, **16**(6), 1082–1089.
- Jin, J., He, K., Tang, X., Zhe, L., Lv, L., and Zhao, Y. 2015. An Arabidopsis transcriptional regulatory map reveals distinct functional and evolutionary features of novel transcription factors. *Mol. Biol. Evol.*, **32**, 1767–1773.
- Johnson, D. B. 1977. Efficient algorithms for shortest paths in sparse networks. *J. Assoc. Comput. Mach.(JACM)*, **24**(1), 1–13.
- Johnson, N. 2012. *A visualization of the Hopf fibration*. <http://www.nilesjohnson.net/hopf.html>.
- Josselyn, S. A., Köhler, S., and Frankland, P. W. 2015. Finding the engram. *Nat. Rev. Neurosci.*, **16**, 521–534.
- Jungck, J. R., Wagner, R., van Loo, D., Grossman, B., Khiripet, N., Khiripet, J., Khantuwan, W., and Hagan, M. 2019. Art forms in nature: radiolaria from Haeckel and Blaschka to 3D nanotomography, quantitative image analysis, evolution, and contemporary art. *Theory in Biosci.*, **138**, 157–187.
- Junier, I., and Rivoire, O. 2016. Conserved units of co-expression in bacterial genomes: an evolutionary insight into transcriptional regulation. *PLoS One*, **11**(5), e0155740.
- Junttila, T., and Kaski, P. 2007. Engineering an efficient canonical labeling tool for large and sparse graphs. SIAM. Pages 135–149.
- Kamei, H., and Cock, P. J. A. 2013a. Computation of balanced equivalence relations and their lattice for a coupled cell network. *SIAM J. Appl. Dyn. Sys.*, **12**(1), 352–382.

- Kamei, H., and Cock, P. J. A. 2013b. Computation of balanced equivalence relations and their lattice for a coupled cell network. *SIAM J. Appl. Dyn. Syst.*, **12**(1), 352–382.
- Kandel, E. R., Markram, H., Matthews, P. M., Yuste, R., and Koch, C. 2013. Neuroscience thinks big (and collaboratively). *Nat. Rev. Neurosci.*, **14**(9), 659–664.
- Kanehisa, M., Goto, S., Kawashima, S., Okuno, Y., and Hattori, M. 2004. The KEGG resource for deciphering the genome. *Nucleic Acids Res.*, **32**(suppl_1), D277–D280.
- Kanehisa, M., Sato, Y., Kawashima, M., Furumichi, M., and Tanabe, M. 2015. KEGG as a reference resource for gene and protein annotation. *Nucleic Acids Res.*, **44**, D457–D462.
- Kanehisa, M., Furumichi, M., Sato, Y., Kawashima, M., and Ishiguro-Watanabe, M. 2023. KEGG for taxonomy-based analysis of pathways and genomes. *Nucleic Acids Res.*, **51**, D587–D592.
- Kanter, I., Kopelowitz, E., Vardi, R., Zigzag, M., Kinzel, W., Abeles, M., and Cohen, D. 2011. Nonlocal mechanism for cluster synchronization in neural circuits. *EPL (Europhys. Lett.)*, **93**(6), 66001.
- Kaplan, S., Bren, A., Zaslaver, A., Dekel, E., and Alon, U. 2008. Diverse two-dimensional input functions control bacterial sugar genes. *Mol. Cell*, **29**(6), 786–792.
- Karlebach, G., and Shamir, R. 2008. Modelling and analysis of gene regulatory networks. *Nat. Rev. Mol. Cell Biol.*, **9**(10), 770–780.
- Karp, P. D., Ong, W. K., Paley, S., Billington, R., Caspi, R., Fulcher, C., Kothari, A., Krummenacker, M., Latendresse, M., Midford, P. E., et al. 2018. The ecocyc database. *EcoSal. Plus*, **8**(1).
- Kato, S., Kaplan, H. S., Schrödel, T., Skora, S., Lindsay, T. H., Yemini, E., Lockery, S., and Zimmer, M. 2015. Global brain dynamics embed the motor command sequence of *Caenorhabditis elegans*. *Cell*, **163**(3), 656–669.
- Kavitha, T., Liebchen, C., Mehlhorn, K., Michail, D., Rizzi, R., Ueckerdt, T., and Zweig, K. A. 2009. Cycle bases in graphs characterization, algorithms, complexity, and applications. *Comp. Sci. Rev.*, **3**(4), 199–243.
- Kelley, J. L. 1955. *General Topology*. Van Nostrand.
- Kelly, P. J. 1957. A congruence theorem for trees. *Pacific J. Math.*, **7**, 961–968.
- Keseler, I. M., Gama-Castro, S., Mackie, A., Billington, R., Bonavides-Martínez, C., Caspi, R., Kothari, A., Krummenacker, M., Midford, P. E., Muñiz-Rascado, L., Ong, W. K., Paley, S., Santos-Zavaleta, A., Subhraveti, P., Tierrafría, V. H., Wolfe, A. J., Collado-Vides, J., Paulsen, I. T., and Karp, P. D. 2021. The EcoCyc Database in 2021. *Front. Microbiol.*, **12**, 711077.
- Khalil, Ahmad S., and Collins, James J. 2010. Synthetic biology: applications come of age. *Nature Rev. Genetics*, **11**(5), 367–379.
- Kibble, T. W. B., and Berkshire, F. H. 2004. *Classical Mechanics*. Imperial College Press.
- Kim, M., Rai, N., Zorraquino, V., and Tagkopoulos, I. 2016. Multi-omics integration accurately predicts cellular state in unexplored conditions for *Escherichia coli*. *Nat. Commun.*, **7**(1), 1–12.
- Kincaid, R. K., and Phillips, D. J. 2011. Network topology measures. *Wiley Interdisciplinary Reviews: Computational Statistics*, **3**(6), 557–565.

- Kitsak, M., Gallos, L. K., Havlin, S., Liljeros, F., Muchnik, L., Stanley, H. E., and Makse, H. A. 2010. Identification of influential spreaders in complex networks. *Nat. Phys.*, **6**(11), 888–893.
- Klein, F. 1872. Vergleichende Betrachtungen über neuere geometrische Forschungen (A comparative review of recent researches in geometry). *Math. Annalen*, **43**, 63–100.
- Klickstein, I., Pecora, L. M., and Sorrentino, F. 2019. Symmetry induced group consensus. *Chaos*, **29**(7), 073101.
- Klipp, E., Liebermeister, W., Wierling, C., and Kowald, A. 2016. *Systems Biology: A textbook*. Wiley.
- Koch, M. A., Norris, D. G., and Hund-Georgiadis, M. 2002. An investigation of functional and anatomical connectivity using magnetic resonance imaging. *Neuroimage*, **16**, 241–250.
- Koch, T., Berthold, T., Pedersen, J., and Vanaret, C. 2022. Progress in mathematical programming solvers from 2001 to 2020. *EURO J. Comp. Optimization*, **10**, 100031.
- Kolesnikov, N., Hastings, E., Keays, M., Melnichuk, O., Tang, Y. A., Williams, E., Dylag, M., Kurbatova, N., Brandizi, M., Burdett, T., et al. 2015. ArrayExpress update—simplifying data submissions. *Nucleic Acids Res.*, **43**(D1), D1113–D1116.
- Kondev, J. 2014. Bacterial decision making. *Physics Today*, **67**(2).
- Kondo, S., and Asai, R. 1995. A reaction-diffusion wave on the skin of the marine angelfish *Pomacanthus*. *Nature*, **376**, 765–768.
- Kossinets, G. 2006. Effects of missing data in social networks. *Social Netw.*, **28**(3), 247–268.
- Kramer, B. P., and Fussenegger, M. 2005. Hysteresis in a synthetic mammalian gene network. *Proc. Natl. Acad. Sci. USA.*, **102**, 9517–9522.
- Kramer, B. P., Viretta, A. U., Baba, M.D.-E., Aube, D., Weber, W., and Fussenegger, M. 2004. An engineered epigenetic transgene switch in mammalian cells. *Nat. Biotech.*, **22**, 867–870.
- Kudose, S. 2009. Equitable partitions and orbit partitions. *Acta Math. Sinica*, 1–9.
- Kupka, I. 1963. Contribution à la théorie des champs génériques. *Contrib. Diff. Eqs.*, **2**, 457–484.
- Kupka, I. 1964. Contribution à la théorie des champs génériques. *Contrib. Diff. Eqs.*, **3**, 411–420.
- Kuramoto, Y. 1984. *Chemical Oscillations, Waves, and Turbulence*. Springer.
- Lachaux, J.-P., Rodriguez, E., Martinerie, J., and Varela, F. J. 1999. Measuring phase synchrony in brain signals. *Hum. Brain Mapp.*, **8**(4), 194–208.
- Lancaster, P., and Tismenetsky, M. 1985. *The Theory of Matrices*. Academic Press.
- Landau, L. D., and Lifshitz, E. M. 1977. *Quantum Mechanics: Non-Relativistic Theory*. Vol. 3. Butterworth-Heinemann.
- Langfelder, P., and Horvath, S. 2008. WGCNA: an R package for weighted correlation network analysis. *BMC Bioinformatics*, **9**(1), 1–13.
- LaSalle, J. P. 1964. Recent advances in Liapunov stability theory. *SIAM Rev.*, **6**, 1–11.
- LaSalle, J. P., and Lefschetz, S. 1961. *Stability by Lyapunov's Second Method with Applications*. Academic Press.

- Lei, E. S., et al. 2007. Genome-wide atlas of gene expression in the adult mouse brain. *Nature*, **445**, 168–176.
- Leifer, I. 2022. *Symmetry-Inspired Analysis of Biological Networks*. City University of New York. PhD Thesis.
- Leifer, I., Morone, F., Reis, S. D. S., Andrade Jr., J. S., and Makse, H. A. 2020. Circuits with broken fibration symmetries perform core logic computations in biological networks. *PLoS Comp. Biol.*, **16**, 1007776.
- Leifer, I., Sánchez-Pérez, M., Ishida, C., and Makse, H.A. 2021. Predicting synchronized gene coexpression patterns from fibration symmetries in gene regulatory networks in bacteria. *BMC Bioinformatics*, **22**, 363.
- Leifer, I., Phillips, D., Sorrentino, F., and Makse, H. A. 2022. Symmetry-driven network reconstruction through pseudobalanced coloring optimization. *J. Stat. Mech.: Theo. and Exp.*, **2022**, 073403.
- Leite, M. C. A., and Golubitsky, M. 2006. Homogeneous three-cell networks. *Nonlinearity*, **19**, 2313–2363.
- Leon, M., Woods, M. L., Fedorec, A. J. H., and Barnes, C. P. 2016. A computational method for the investigation of multistable systems and its application to genetic switches. *BMC Syst. Biol.*, **10**(1), 1–12.
- Li, L., Alderson, D., Doyle, J. C., and Willinger, W. 2005. Towards a theory of scale-free graphs: Definition, properties, and implications. *Internet Math.*, **2**(4), 431–523.
- Li, Q., Dong, J. W., Del Ferraro, G., Petrovich Brennan, N., Peck, K. K., Tabar, V., Makse, H. A., and Holodny, A. I. 2019. Functional translocation of Broca’s area in a low-grade left frontal glioma: graph theory reveals the novel, adaptive network connectivity. *Front. Neurol.*, **10**, 702.
- Li, Q., Del Ferraro, G., Pasquini, L., Peck, K. K., Makse, H. A., and Holodny, A. I. 2020. Core language brain network for fMRI language task used in clinical applications. *Network Neurosci.*, **4**(1), 134–154.
- Li, Q., Pasquini, L., Del Ferraro, G., Gene, M., Peck, K. K., Makse, H. A., and Holodny, A. I. 2021. Monolingual and bilingual language networks in healthy subjects using functional MRI and graph theory. *Sci. Rep.*, **11**(1), 10568.
- Liang, J., Gurukar, S., and Parthasarathy, S. 2021. MILE: A Multi-Level Framework for Scalable Graph Embedding. *Proceedings of the International AAAI Conference on Web and Social Media*, **15**(May), 361–372.
- Liao, J. C., Boscolo, R., Yang, Y.-L., Tran, L. M., Sabatti, C., and Roychowdhury, V. P. 2003. Network component analysis: reconstruction of regulatory signals in biological systems. *Proc. Natl. Acad. Sci. USA*, **100**(26), 15522–15527.
- Liapunov, A. M. 1893. Obshchaya Zadacha Ustoichivosti Drizheniya. *Comm. Soc. Math. Kharkov*, **3**, 265–272.
- Liben-Nowell, D., and Kleinberg, J. 2007. The link-prediction problem for social networks. *J. Amer. Soc. Information Sci. Tech.*, **58**(7), 1019–1031.
- Lin, Tianyang, Wang, Yuxin, Liu, Xiangyang, and Qiu, Xipeng. 2021 (June). *A Survey of Transformers*. arXiv:2106.04554 [cs].
- Linton, S. 2007. GAP: groups, algorithms, programming. *ACM Commun. Comp. Alg.*, **41**(3), 108–109.

- Lodi, M., Sorrentino, F., and Storace, M. 2021. One-way dependent clusters and stability of cluster synchronization in directed networks. *Nature Commun.*, **12**(1), 4073.
- Lodi, M., Panahi, S., Sorrentino, F., Torcini, A., and Storace, M. 2024. Patterns of synchronized clusters in adaptive networks. *Commun. Phys.*, **7**(1), 198.
- Lü, L., and Zhou, T. 2011. Link prediction in complex networks: A survey. *Physica A*, **390**(6), 1150–1170.
- Luks, E. M. 1982. Isomorphism of graphs of bounded valance can be tested in polynomial time. *J. Comput. Syst. Sci.*, 42–65.
- Luks, E. M. 1993. Permutations groups and polynomial-time computation. Pages 139–175 of: *DIMACS Series in Discrete Mathematics and Theoretical Computer Science (vol 11)*. AMS.
- Ma, H., Buer, J., and Zeng, A. 2004. Hierarchical structure and modules in the *Escherichia coli* transcriptional regulatory network revealed by a new top-down approach. *BMC bioinformatics*, **5**(1), 1–10.
- Mac Lane, S. 2013. *Categories for the Working Mathematician*. Vol. 5. Springer Science & Business Media.
- MacArthur, B. D., Sánchez-García, R. J., and Anderson, J. W. 2008. Symmetry in complex networks. *Discrete Appl. Math.*, **156**(18), 3525–3531.
- Macía, J., Widder, S., and Solé, R. 2009. Specialized or flexible feed-forward loop motifs: a question of topology. *BMC Syst. Biol.*, **3**(1), 1–18.
- Maertens, A., Tran, V., Kleensang, A., and Hartung, T. 2018. Weighted Gene Correlation Network Analysis (WGCNA) reveals novel transcription factors associated with bisphenol a dose-response. *Front. Genet.*, **9**, 508.
- Makse, H. A., and Zava, M. 2025. *The Science of Influencers and Superspreaders: Using Networks and Artificial Intelligence to Understand Fake News, Pandemics, Markets, and the Brain*. Springer.
- Maldacena, J. 2015. The symmetry and simplicity of the laws of physics and the Higgs boson. *Euro. J. Phys.*, **37**(1), 015802.
- Mangan, S., and Alon, U. 2003. Structure and function of the feed-forward loop network motif. *Proc. Natl. Acad. Sci. USA*, **100**(21), 11980–11985.
- Mangan, S., Zaslaver, A., and Alon, U. 2003. The coherent feedforward loop serves as a sign-sensitive delay element in transcription networks. *J. Mol. Biol.*, **334**(2), 197–204.
- Marbach, D., Costello, J. C., Küffner, R., Vega, N. M., Prill, R. J., Camacho, D. M., Allison, K. R., Kellis, M., Collins, J. J., and Stolovitzky, G. 2012. Wisdom of crowds for robust gene network inference. *Nat. Methods*, **9**(8), 796–804.
- Martínez-Antonio, A., Salgado, H., Gama-Castro, S., Gutiérrez-Ríos, R. M., Jiménez-Jacinto, V., and Collado-Vides, J. 2003. Environmental conditions and transcriptional regulation in *Escherichia coli*: a physiological integrative approach. *Biotech & Bioeng.*, **84**(7), 743–749.
- Martínez-Antonio, A., Janga, S. C., and Thieffry, D. 2008. Functional organisation of *Escherichia coli* transcriptional regulatory network. *J. Molec. Biol.*, **381**(1), 238–247.
- Mastrandrea, R., Cecchetti, L., Lettieri, G., Handjaras, G., Leo, A., Papale, P., Gili, T., Martini, N., Latta, D. D., Chiappino, D., et al. 2023. Information load dynamically

- modulates functional brain connectivity during narrative listening. *Sci. Rep.*, **13**(1), 8110.
- Mayo, A. E., Setty, Y., Shavit, S., Zaslaver, A., and Alon, U. 2006. Plasticity of the cis-regulatory input function of a gene. *PLoS Biol.*, **4**(4), e45.
- McKay, B. D. 1981. Practical graph isomorphism. *Congressus Numerantium*, **30**, 45–87.
- McKay, B. D. 1990. *Nauty User's Guide (Version 1.5)* Computer Science Department, Australian National University. Tech. rept. Canberra, Tech. Rep. TR-CS-90-02.
- McKay, B. D., and Piperno, A. 2014. Practical graph isomorphism, II. *J. Symbolic Comp.*, **60**, 94–112.
- Medina-Rivera, A., Abreu-Goodger, C., Thomas-Chollier, M., Salgado, H., Collado-Vides, Julio., and van Helden, J. 2011. Theoretical and empirical quality assessment of transcription factor-binding motifs. *Nucleic Acids Res.*, **39**(3), 808–824.
- Meinhardt, H. 1995. *The Algorithmic Beauty of Seashells*. Springer-Verlag, Berlin.
- Menara, T., Baggio, G., Bassett, D. S., and Pasqualetti, F. 2019. Stability conditions for cluster synchronization in networks of heterogeneous Kuramoto oscillators. *arXiv*, 1806.06083v2.
- Meshulam, L., Gauthier, J. L., Brody, C. D., Tank, D. W., and Bialek, W. 2019. Coarse graining, fixed points, and scaling in a large population of neurons. *Phys. Rev. Lett.*, **123**(17), 178103.
- Métris, A., Sudhakar, P., Fazekas, D., Demeter, A., Ari, E., Olbei, M., Branchu, P., Kingsley, A., Baranyi, J., and Korcsmaros, T. 2017. SalmoNet, an integrated network of ten *Salmonella enterica* strains reveals common and distinct pathways to host adaptation. *NPJ Systems Biol. & Appl.*, **3**, 31.
- Millán, A. P., Sun, H., Giambagli, L., Muolo, R., Carletti, T., Torres, J. J., Radicchi, F., Kurths, J., and Bianconi, G. 2025. Topology shapes dynamics of higher-order networks. *Nat. Phys.*
- Milo, R., Shen-Orr, S., Itzkovitz, S., Kashtan, N., Chklovskii, D., and Alon, U. 2002. Network motifs: simple building blocks of complex networks. *Science*, **298**(5594), 824–827.
- Monod, J. 1970. Symmetry and Function of Biological Systems at the Macromolecular Level. In: Engström, A., and Strandberg, B. (eds), *Proc. 11th Nobel Symposium, Södergarn, Lidingö, Sweden, Aug. 1968*. Interscience (Wiley) and Almqvist and Wiksell.
- Monteiro, H. S., Leifer, I., Reis, S. D. S., Andrade Jr, J. S., and Makse, H. A. 2022. Efficient algorithmic paradigm to identify partial synchrony in information-processing networks using graph fibrations. *Chaos*, **32**, 033120.
- Moretto, M., Sonego, P., Dierckxsens, N., Brilli, M., Bianco, L., Ledezma-Tejeida, D., Gama-Castro, S., Galardini, M., Romualdi, C., Laukens, K., et al. 2016. COLOMBOS v3. 0: leveraging gene expression compendia for cross-species analyses. *Nucleic Acids Res.*, **44**(D1), D620–D623.
- Moriyasu, K. 1983. *An Elementary Primer for Gauge Theory*. World Scientific.
- Morone, F., and Makse, H. A. 2015. Influence maximization in complex networks through optimal percolation. *Nature*, **524**, 65–68.

- Morone, F., and Makse, H. A. 2019. Symmetry group factorization reveals the structure-function relation in the neural connectome of *Caenorhabditis elegans*. *Nature Comm.*, **10**, 4961.
- Morone, F., Leifer, I., and Makse, H. A. 2020. Fibration symmetries uncover the building blocks of biological networks. *Proc. Natl. Acad. Sci. USA*, **117**(15), 8306–8314.
- Morris, C., and Lecar, H. 1981. Voltage oscillations in the barnacle giant muscle fiber. *Biophys. J.*, **35**(1), 193–213.
- Mulas, R., Sánchez-García, R. J., and MacArthur, B. D. 2020. Hypergraph Automorphisms. *arXiv*, 2010.01049.
- Müller-Hill, B. 1996. *The lac Operon: A Short History of a Genetic Paradigm*. Berlin, New York: De Gruyter.
- Munkres, J. R. 2018. *Elements of Algebraic Topology*. CRC press.
- Murray, J. 1989. *Mathematical Biology*. Springer.
- Murray, R. M., Li, Z., and Sastry, S. S. 1993. *A Mathematical Introduction to Robotic Manipulation*. CRC Press.
- Nagumo, J., Arimoto, S., and Yoshizawa, S. 1962. An active pulse transmission line simulating nerve axon. *Proc. IRE*, **50**(10), 2061–2070.
- Nathe, C., Huang, K., Lodi, M., Storace, M., and Sorrentino, F. 2020. Delays induced cluster synchronization in chaotic networks. *Chaos*, **30**(12).
- Newman, M. 2018. *Networks*. Oxford University Press.
- Newman, M. E. J. 2001. Clustering and preferential attachment in growing networks. *Phys. Rev. E*, **64**(2), 025102.
- Newton, I. 1833. *Philosophiae Naturalis Principia Mathematica*. Vol. 2. typis A. et JM Duncan.
- Nguyen, J. P., Shipley, F. B., Linder, A. N., Plummer, G. S., Liu, M., Setru, S. U., Shaevitz, J. W., and Leifer, A. M. 2016. Whole-brain calcium imaging with cellular resolution in freely behaving *Caenorhabditis elegans*. *Proc. Natl. Acad. Sci. USA*, **113**(8), E1074–E1081.
- Nichols, A. L. A., Eichler, T., Latham, R., and Zimmer, M. 2017. A global brain state underlies *C. elegans* sleep behavior. *Science*, **356**(6344), eaam6851.
- Nijholt, E., Rink, R., and Sanders, J. 2016. Graph fibrations and symmetries of network dynamics. *J. Diff. Eq.*, **261**(9), 4861–4896.
- Nishikawa, T., and Motter, A. E. 2016. Network-complement transitions, symmetries, and cluster synchronization. *Chaos*, **26**(9), 094818.
- Nishikawa, T., and Motter, A. E. 2010. Network synchronization landscape reveals compensatory structures, quantization, and the positive ef. *Proc. Natl. Acad. Sci. USA*, **107**, 10342–10347.
- Norris, N. 1995. Universal covers of graphs: isomorphism to depth $n - 1$ implies isomorphism to all depths. *Discrete Appl. Math.*, **56**(1), 61–74.
- Nykamp, D. Q. 2024. The idea of synchrony of phase oscillators. *Math. Insight*, **60**, mathinsight.org/synchrony_phase_oscillators_idea.
- Oh, S. W., et al. 2014. A mesoscale connectome of the mouse brain. *Nature*, **508**, 207–214.
- Ostermeier, L. 2015. *(Relaxed) Product Structures of Graphs and Hypergraphs*. Ph.D. thesis, Universität Leipzig.

- Ott, E. 2002. *Chaos in dynamical systems*. Cambridge university press.
- Paige, R., and Tarjan, R. E. 1987. Three partition refinement algorithms. *SIAM J.Comput.*, **16**(6), 973–989.
- Panahi, S., Klickstein, I., and Sorrentino, F. 2021. Cluster synchronization of networks via a canonical transformation for simultaneous block diagonalization of matrices. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, **31**(11).
- Park, H.-J., and Friston, K. 2013. Structural and functional brain networks: from connections to cognition. *Science*, **342**(6158), 1238411.
- Payne, J. L., and Wagner, A. 2015. Function does not follow form in gene regulatory circuits. *Sci. Rep.*, **5**(1), 1–10.
- Pecora, L. M., and Carroll, T. L. 1990. Synchronization in chaotic systems. *Phys. Rev. Lett.*, **64**, 821–825.
- Pecora, L. M., and Carroll, T. L. 1998. Master stability functions for synchronized coupled systems. *Phys. Rev. Lett.*, **80**(10), 2109.
- Pecora, L. M., Sorrentino, F., Hagerstrom, A. M., Murphy, T. E., and Roy, R. 2014. Cluster synchronization and isolated desynchronization in complex networks with symmetries. *Nat. Comm.*, **5**(1), 1–8.
- Peixoto, M. M. 1966. On an approximation theorem of Kupka and Smale. *J. Diff. Eq.*, **3**, 214–227.
- Phillips, J. M., and Venkatasubramanian, S. 2011. A gentle introduction to the kernel distance. *arXiv*, 1103.1625.
- Pikovsky, A., Rosenblum, M., and Kurths, J. 2001. *Synchronization: a Universal Concept in Nonlinear Sciences*. Cambridge University Press.
- Pisanski, T., and Vrabec, J. 1983. *Graph Bundles*. unpublished.
- Poldrack, R. A. 2021. The physics of representation. *Synthèse*, **199**(1), 1307–1325.
- Polyak, B. T., and Kvinto, Y. I. 2017. Stability and synchronization of oscillators: new Lyapunov functions. *Automation & Remote Control*, **78**, 1234–1242.
- Poo, M. M., Pignatelli, M., Ryan, T. J., Tonegawa, S., Bonhoeffer, T., Martin, K. C., Rudenko, A., Tsai, L.-H., Tsien, R. W., Fishell, G., Mullins, C., Goncalves, J. T., Shtrahman, M., Johnston, S. T., Gage, F. H., Dan, Y., Long, J., Buzsáki, G., and Stevens, C. 2016. What is memory? The present state of the engram. *BMC Biology*, **14**, 40.
- Poston, T., and Stewart, I. 2014. *Catastrophe Theory and Its Applications*. Courier Corporation.
- Pratapa, A., Jalihal, A. P., Law, J. N., Bharadwaj, A., and Murali, T. M. 2020. Benchmarking algorithms for gene regulatory network inference from single-cell transcriptomic data. *Nat. Methods*, **17**(2), 147–154.
- Preti, G., Fazzone, A., and De Francisci Morales, G. 2024. Higher-Order Null Models as a Lens for Social Systems. *Phys. Rev. X*, **14**(3), 031032.
- Purcell, O., Savery, N. J., Grierson, C. S., and Di Bernardo, M. 2010. A comparative analysis of synthetic genetic oscillators. *J. R. Soc. Interface*, **7**(52), 1503–1524.
- Rackham, O., and Chin, J. W. 2005. A network of orthogonal ribosome mRNA pairs. *Nat. Chem. Bio.*, **1**, 159–166.
- Randić, M. 1975. Characterization of molecular branching. *J. Am. Chem. Soc.*, **97**(23), 6609–6615.

- Reis, S. D. S., Hu, Y., Babino, A., Andrade Jr., J. S., Canals, S., Sigman, M., and Makse, H. A. 2014. Avoiding catastrophic failure in correlated networks of networks. *Nat. Phys.*, **10**, 762–767.
- Renier, N., Adams, E. L., Kirst, C., Wu, Z., Azevedo, R., Kohl, J., Autry, A. E., Kadiri, L., Umadevi Venkataraju, K., Zhou, Y., Wang, V. X., Tang, C. Y., Olsen, O., Dulac, C., Osten, P., and Tessier-Lavigne, M. 2016. Mapping of brain activity by automated volume analysis of immediate early genes. *Cell*, **165**, 1789–1802.
- Rink, B., and Sanders, J. 2013. Coupled cell networks and their hidden symmetries. *SIAM J. Math. Anal.*, **46**, 1577–1609.
- Rolls, E. T., and Treves, A. 1998. *Neural Networks and Brain Function*. Oxford University Press.
- Rosin, D. P., Rontani, D., Gauthier, D. J., and Schöll, E. 2013. Control of synchronization patterns in neural-like Boolean networks. *Phys. Rev. Lett.*, **110**(10), 104102.
- Roy, D. S., Park, Y.-G., Kim, M. E., Zhang, Y., Ogawa, S. K., DiNapoli, N., Gu, X., Cho, J. H., Choi, H., Kamensky, L., Martin, J., Mosto, O., Aida, T., Chung, K., and Tonegawa, S. 2022. Brain-wide mapping reveals that engrams for a single memory are distributed across multiple brain regions. *Nat. Commun.*, **13**, 1799.
- Roy, S., Bhattacharyya, D. K., and Kalita, J. K. 2014. Reconstruction of gene co-expression network from microarray data using local expression patterns. *BMC Bioinformatics*, **15**(7), 1–14.
- Rubinov, M., and Sporns, O. 2010. Complex network measures of brain connectivity: uses and interpretations. *Neuroimage*, **52**(3), 1059–1069.
- Russo, G., and Slotine, J.-J. E. 2011. Symmetries, stability, and control in nonlinear systems and networks. *Phys. Rev. E*, **84**(4), 041929.
- Ryan, T. J., Dheeraj, S. R., Pugnatelli, M., Arons, A., and Tonegawa, S. 2015. Engram cells retain memory under retrograde amnesia. *Science*, **348**, 1007–1013.
- Sánchez-García, R. J. 2020. Exploiting symmetry in network analysis. *Commun. Phys.*, **3**(1), 1–15.
- Santos-Zavaleta, A., Sánchez-Párez, M., Salgado, H., Velázquez-Ramírez, D.A., Gama-Castro, S., Tierrafría, V. H., Busby, S. J. W., Aquino, P., Fang, X., Palsson, B. O., Galagan, J. E., and Collado-Vides, J. 2018. A unified resource for transcriptional regulation in *Escherichia coli* K-12 incorporating high-throughput-generated binding data into RegulonDB version 10.0. *BMC Biol.*, **16**(1), 91.
- Schaub, M. T., O’Clery, N., Billeh, Y. N., Delvenne, J.-C., Lambiotte, R., and Barahona, M. 2016. Graph partitions and cluster synchronization in networks of oscillators. *Chaos*, **26**(9), 094821.
- Schoffelen, J.-M., and Gross, J. 2009. Source connectivity analysis with MEG and EEG. *Hum. Brain Mapp.*, **30**(6), 1857–1865.
- Schrödel, T., Prevedel, R., Aumayr, K., Zimmer, M., and Vaziri, A. 2013. Brain-wide 3D imaging of neuronal activity in *Caenorhabditis elegans* with sculpted light. *Nature Methods*, **10**(10), 1013–1020.
- Schrödinger, E. 1944. *What is Life?* Cambridge University Press.
- Schwenk, A. J. 1974. Computing the characteristic polynomial of a graph. Pages 153–172 of: Bari, R. A., and Harary, F. (eds), *Graphs and Combinatorics*. Springer.

- Schwichtenberg, J. 2019. *Physics from Finance: A gentle introduction to gauge theories, fundamental interactions and fiber bundles*. No-Nonsense Books.
- Seifert, H. 1933. Topologie dreidimensionaler gefaserter Räume. *Acta Math.*, **60**, 147–238.
- Seitzman, B. A., Snyder, A. Z., Leuthardt, E. C., and Shimony, J. S. 2019. The state of resting state networks. *Topics Mag. Res. Imaging: TMRI*, **28**(4), 189.
- Semon, R. 1921. *The Mneme*. George Allen & Unwin.
- Sequeira, P., Aguiar, A. P., and Hespanha, J. P. 2021. Commutative monoid formalism for weighted coupled cell networks and invariant synchrony patterns. *SIAM J. Appl. Dyn. Sys.*, **20**, 10.1137/20M1387109.
- Sequeira, P. M., Hespanha, J., and Aguiar, A. P. 2022. Decomposition of admissible functions in weighted coupled cell networks. *arXiv*, 2201.04972v1.
- Serre, J.-P. 1951. Homologie singulière des espaces fibrés. Applications. *Annu. Math.*, **54**, 425–505.
- Shen-Orr, S. S., Milo, R., Mangan, S., and Alon, U. 2002. Network motifs in the transcriptional regulation network of *Escherichia coli*. *Nature Genetics*, **31**(1), 64–68.
- Sheth, R., Marcon, L., Bastida, M.F., Junco, M., Quintana, L., Dahn, R., Kmita, M., Sharpe, J., and Ros, M.A. 2012. Hox genes regulate digit patterning by controlling the wavelength of a Turing-type mechanism. *Science*, **338**, 1476–1480.
- Shimada, T., Ogasawara, H., and Ishihama, A. 2018. Single-target regulators form a minor group of transcription factors in *Escherichia coli* K-12. *Nucleic Acids Res.*, **46**, 3921–3936.
- Sick, S., Reinker, S., Timmer, J., and Schlake, T. 2006. WNT and DKK determine hair follicle spacing through a reaction-diffusion mechanism. *Science*, **314**, 1476–1480.
- Siddique, A. Ba., Pecora, L., Hart, J. D., and Sorrentino, F. 2018. Symmetry and input-cluster synchronization in networks. *Phys. Rev. E*, **97**(4), 042217.
- Sigman, M., and Dehaene, S. 2008. Brain mechanisms of serial and parallel processing during dual-task performance. *J. Neurosci.*, **28**, 7585–7598.
- Singer, W. 1999. Neuronal synchrony: a versatile code for the definition of relations. *Neuron*, **24**, 49–65.
- Skora, S., Mende, F., and Zimmer, M. 2018. Energy scarcity promotes a brain-wide sleep state modulated by insulin signaling in *C. elegans*. *Cell Rep.*, **22**(4), 953–966.
- Sloane, N. J. A. 2020. *The On-Line Encyclopedia of Integer Sequences*. OEIS Foundation Inc.
- Smale, S. 1963. Stable manifolds for differential equations and diffeomorphisms. *Annu. Scuola Norm. Sup. Pisa*, **18**, 214–227.
- Smale, S. 1967. Differentiable dynamical systems. *Bull. Amer. Math. Soc.*, **73**, 747–817.
- Sojoudi, S. 2016. Equivalence of graphical lasso and thresholding for sparse graphs. *J. Mach. Learn. Res.*, **17**, 1–21.
- Sorrentino, F., and Ott, E. 2007. Network synchronization of groups. *Phys. Rev. E*, **76**(5), 056114.
- Sorrentino, F., Pecora, L. M., Hagerstrom, A. M., Murphy, T. E., and Roy, R. 2016. Complete characterization of the stability of cluster synchronization in complex dynamical networks. *Sci. Advances*, **2**(4), e1501737.

- Sorrentino, Francesco, and Pecora, Louis. 2016. Approximate cluster synchronization in networks with symmetries and parameter mismatches. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, **26**(9).
- Steenrod, N. 2016. *The Topology of Fibre Bundles*. Princeton University Press.
- Stein, W., et al. 2006. *SAGE: Software for Algebra and Geometry Experimentation*. <http://www.sagemath.org>.
- Stephens, G. J., Johnson-Kerner, B., Bialek, W., and Ryu, W. S. 2008. Dimensionality and dynamics in the behavior of *C. elegans*. *PLoS Comp. Biol.*, **4**(4), e1000028.
- Stephens, G. J., Johnson-Kerner, B., Bialek, W., and Ryu, W. S. 2010. From modes to movement in the behavior of *Caenorhabditis elegans*. *PloS One*, **5**(11), e13914.
- Stephens, G. J., Osborne, L. C., and Bialek, W. 2011. Searching for simplicity in the analysis of neurons and behavior. *Proc. Natl. Acad. Sci. USA*, **108**, 15565–15571.
- Stewart, I. 2007. The lattice of balanced equivalence relations of a coupled cell network. *Math. Proc. Camb. Philos. Soc.*, **143**(1), 165.
- Stewart, I. 2022. Symmetry and network architecture in neuronal circuits: Complicity of form and function. *Internat. J. Bif. Chaos*, **32**, 2230033.
- Stewart, I. 2024. Groupoids, fibrations, and balanced colorings of networks. *Internat. J. Bif. Chaos*, **34**, 2430014.
- Stewart, I., and Wood, D. 2024. Stable synchronous propagation of periodic signals by feedforward networks. *SIAM J. Appl. Dyn. Sys.*, **23**, doi: 10.1137/23M1552267.
- Stewart, I., and Wood, D. 2025a. Stable synchronous propagation in feedforward networks for biped locomotion. *in preparation*.
- Stewart, I., and Wood, D. 2025b. Stable synchronous propagation of signals in feedforward networks of standard model neurons. *in preparation*.
- Stewart, I., Golubitsky, M., and Pivato, M. 2003. Symmetry groupoids and patterns of synchrony in coupled cell networks. *SIAM J. Appl. Dyn. Sys.*, **2**(4), 609–646.
- Stewart, I., Reis, S. D. S., and Makse, H. A. 2024. Dynamics and bifurcations in genetic circuits with fibration symmetries. *J. R. Soc. Interface*, **21**, 20240386.
- Stewart, I. N., and Tall, D. O. 2015. *The Foundations of Mathematics*. Oxford University Press.
- Stivala, A., and Lomi, A. 2021. Testing biological network motif significance with exponential random graph models. *Appl. Network Sci.*, **6**:91, doi.org/10.1007/s41109-021-00434-y.
- Stricker, J., Cookson, S., Bennett, M. R., Mather, W. H., Tsimring, L. S., and Hasty, J. 2008. A fast, robust and tunable synthetic gene oscillator. *Nature*, **456**(7221), 516–519.
- Strogatz, S. H. 2018. *Nonlinear Dynamics and Chaos*. CRC Press.
- Tabor, J. J., Salis, H., Simpson, Z. B., Chevalier, A. A., Levskaya, A., Marcotte, E. M., Voigt, C. A., and Ellington, A. D. 2009. A synthetic genetic edge detection program. *Cell*, **137**, 1272–1281.
- Tanenbaum, A. S. 2016. *Structured Computer Organization*. Pearson Education India.
- Teghipco, A., Hussain, A., and Tivarus, M. E. 2016. Disrupted functional connectivity affects resting state based language lateralization. *Neuroimage*, **12**, 910–927.

- Tegner, J., Yeung, M. K., Hasty, J., and Collins, J. J. 2003. Reverse engineering gene networks: integrating genetic perturbations with dynamical modeling. *Proc. Natl. Acad. Sci. USA*, **100**(10), 5944–5949.
- Thom, R. 1975. *Structural Stability and Morphogenesis*. Benjamin.
- Thompson, D. W. 1942. *On Growth and Form*. Cambridge University Press.
- Thut, G., Miniussi, C., and Gross, J. 2012. The functional importance of rhythmic activity in the brain. *Current Biol.*, **22**(16), R658–R663.
- Tian, Q., Zou, J., Tang, J., Liang, L., Cao, X., and Fan, S. 2022. scMelody: an enhanced consensus-based clustering model for single-cell methylation data by reconstructing cell-to-cell similarity. *Front. Bioeng. and Biotech.*, **10**, 842019.
- Tibshirani, R. 1996. Regression shrinkage and selection via the lasso. *J. Royal Stat. Soc. Ser. B*, **58**, 267–288.
- Tierrafría, V. H., Rioualen, C., Salgado, H., Lara, P., Gama-Castro, S., Lally, P., Gómez-Romero, L., Peña-Loredo, P., et al. 2022. RegulonDB 11.0: Comprehensive high-throughput datasets on transcriptional regulation in Escherichia coli K-12. *Microbial Genomics*, **8**(5), 000833.
- Tigges, M., Marquez-Lago, T. T., Stelling, J., and Fussenegger, M. 2009. A tunable synthetic mammalian oscillator. *Nature*, **457**(7227), 309–312.
- Tononi, G., Sporns, O., and Edelman, G. M. 1994. A measure for brain complexity: relating functional segregation and integration in the nervous system. *Proc. Natl. Acad. Sci. USA*, **91**(11), 5033–5037.
- Transtrum, M. K., Machta, B. B., Brown, K. S., Daniels, B. C., Myers, C. R., and Sethna, J. P. 2015. Perspective: Sloppiness and emergent theories in physics, biology, and beyond. *J. Chem. Phys.*, **143**(1), 07B201_1.
- Turing, A. M. 1952. The chemical basis of morphogenesis. *Phil. Trans. Roy. Soc. London B*, **237**(37).
- Tutukina, M. N., Potapova, A. V., Vlasov, P. K., Purtov, Y. A., and Ozoline, O. N. 2016. Structural modeling of the ExuR and UxuR transcription factors of *E. coli*: search for the ligands affecting their regulatory properties. *J. Biomol. Struct. Dyn.*, **34**, 2296–304.
- Tyson, J. J., Chen, K. C., and Novak, B. 2003. Sniffers, buzzers, toggles and blinkers: dynamics of regulatory and signaling pathways in the cell. *Current Opinion Cell Biol.*, **15**(2), 221–231.
- Uhlhaas, P. J., Pipa, G., Lima, B., Melloni, L., Neuenschwander, S., Nikolić, D., and Singer, W. 2009. Neural synchrony in cortical networks: history, concept and current status. *Front. Integr. Neurosci.*, **100**.
- Unger, S. H. 1964. GIT—A heuristic program for testing pairs of directed line graphs for isomorphism. *Comm. ACM*, **7**(1), 26–34.
- Uzel, K., Kato, S., and Zimmer, M. 2022. A set of hub neurons and non-local connectivity features support global brain dynamics in *C. elegans*. *Current Biology*, **32**, 3443–3459.
- Vainshtein, B. K. 1994. *Modern Crystallography, Vol. 1. Fundamentals of Crystals. Symmetry, and Methods of Structural Crystallography*. Springer.
- van den Heuvel, M. P., and Sporns, O. 2013. Network hubs in the human brain. *Trends Cognit. Sci.*, **17**(12), 683–696.

- van den Heuvel, M. P., and Sporns, O. 2019. A cross-disorder connectome landscape of brain dysconnectivity. *Nat. Rev. Neurosci.*, **20**(7), 435–446.
- van den Heuvel, M. P., Mandl, R. C. W., Kahn, R. S., and Hulshoff Pol, H. E. 2009. Functionally linked resting-state networks reflect the underlying structural connectivity architecture of the human brain. *Hum. Brain Mapp.*, **30**, 3127–3141.
- VanElzakker, M., Fevuraly, R. D., Breindel, T., and L., Spencer. R. 2008. Environmental novelty is associated with a selective increase in Fos expression in the output elements of the hippocampal formation and the perirhinal cortex. *Learn. Mem.*, **15**, 899–908.
- Vann, S. D., Brown, M. W., Erichsen, J. T., and Aggleton, J. P. 2000. Using fos imaging in the rat to reveal the anatomical extent of the disruptive effects of fornix lesions. *J. Neurosci.*, **20**, 2711–2718.
- Varshney, L. R., Chen, B. L., Paniagua, E., Hall, D. H., and Chklovskii, D. B. 2011. Structural properties of the *Caenorhabditis elegans* neuronal network. *PLoS Comput. Biol.*, **7**(2), e1001066.
- Velarde, O., Parra, L., Boldi, P., and Makse, H. A. 2025. The role of fibration symmetries in Geometric Deep Learning. *arXiv*, 2408.1589.
- Von der Gracht, S., Nijholt, E., and Rink, B. 2023. Hypernetworks: cluster synchronization is a higher-order effect. *arXiv*, 2302.08974v2.
- Wang, Y. X. R., and Huang, H. 2014. Review on statistical methods for gene network reconstruction using expression data. *J. Theoret. Biol.*, **362**, 53–61.
- Watson, J. D., and Crick, F. H. 1958. On protein synthesis. Pages 138–163 of: *The Symposia of the Society for Experimental Biology*, vol. 12.
- Watts, D. J., and Strogatz, S. H. 1998. Collective dynamics of ‘small-world’ networks. *Nature*, **393**(6684), 440–442.
- Weinberg, S. 1995. *The Quantum Theory of Fields*. Vol. 2. Cambridge University Press.
- Weisfeiler, B., and Leman, A. 1968. The reduction of a graph to canonical form and the algebra which appears therein. *Nauchno-Technicheskaya Informatsia*, **2**(9), 12–16.
- Weyl, H. 1919. A new extension of the theory of relativity. *Annu. Physik*, **59**, 101.
- Weyl, H. 1929a. Elektron und gravitation. I. *Zeitschrift für Physik*, **56**(5-6), 330–352.
- Weyl, H. 1929b. Gravitation and the electron. *Proc. Natl. Acad. Sci. USA*, **15**(4), 323.
- White, J. G., Southgate, E., Thomson, J. N., and Brenner, S. 1986. The structure of the nervous system of the nematode *Caenorhabditis elegans*. *Phil. Trans. R Soc. Lond. B*, **314**(1165), 1–340.
- Whitney, H. 1935. Sphere spaces. *Proc. Natl. Acad. Sci. USA*, **21**, 462–468.
- Widlar, R. J. 1967. Low-value current source for integrated circuits. *US Patent Number 3,320,439*. Filed May 26, 1965, Granted May 16, 1967.
- Widlar, R. J. 1969a. Design techniques for monolithic operational amplifiers. *IEEE Solid-State Circuits*, **4**, 184–191.
- Widlar, R. J. 1969b. Some circuit design techniques for linear integrated circuits. *IEEE Trans. Circuit Theory*, **4**, 586–590.
- Wikipedia. 2022. *Wikimedia Foundation, Frucht graph*. https://en.wikipedia.org/wiki/Frucht_graph [Accessed Aug 2022].
- Wilczek, F. 2005. In search of symmetry lost. *Nature*, **433**(7023), 239–247.
- Wilczek, F. 2016. *A Beautiful Question: Finding Nature’s Deep Design*. Penguin.

- Wilson, K. G. 1974. Confinement of quarks. *Phys. Rev. D*, **10**(8), 2445.
- Winkler, A. M., Ridgway, G. R., Webster, M. A., Smith, S. M., and Nichols, T. E. 2014. Permutation inference for the general linear model. *Neuroimage*, **92**, 381–397.
- Winkler, A. M., Ridgway, G. R., Douaud, G., Nichols, T. E., and Smith, S. M. 2016. Faster permutation inference in brain imaging. *Neuroimage*, **141**, 502–516.
- Wuppuluri, S., and Stewart, I. (eds). 2022. *From Electrons to Elephants and Elections: Exploring the role of content and context*. Springer, Frontiers Collection.
- Xu, Ke., Hu, W., Leskovec, J., and Jegelka, S. 2018. How powerful are graph neural networks? *arXiv:1810.00826*.
- Yan, G., Vértes, P. E., Towson, E. K., Chew, Y. L., Walker, D. S., Schafer, W. R., and Barabási, A.-L. 2017. Network control principles predict neuron function in the *Caenorhabditis elegans* connectome. *Nature*, **550**(7677), 519–523.
- Yang, T.-H., Wang, C.-C., Wang, Y.-C., and Wu, W.-S. 2014. YTRP: a repository for yeast transcriptional regulatory pathways. *Database*, **2014**, bau014.
- Yemini, E., Lin, A., Nejatbakhsh, A., Varol, E., Sun, R., Mena, G. E., Samuel, A. D. T., Paninski, L., Venkatachalam, V., and Hobert, O. 2021. NeuroPAL: a multicolor atlas for whole-brain neuronal identification in *C. elegans*. *Cell*, **184**(1), 272–288.
- Young, K. 1999. Foreign exchange market as a lattice gauge theory. *Amer. J. Phys.*, **67**(10), 862–868.
- Yu, G., Li, F., Qin, Y., Bo, X., Wu, Y., and Wang, S. 2010. GOSemSim: an R package for measuring semantic similarity among GO terms and gene products. *Bioinformatics*, **26**(7), 976–978.
- Yuste, R., and Church, G. M. 2014. The new century of the brain. *Scientific American*, **310**(3), 38–45.
- Zeeman, E.C. 1977. *Catastrophe Theory: Selected Papers 1972-1977*. Addison-Wesley.
- Zelcbuch, L., Antonovsky, N., Bar-Even, A., Levin-Karp, A., et al. 2013. Spanning high-dimensional expression space using ribosome-binding site combinatorics. *Nucleic Acids Res.*, **41**(9), e98–e98.
- Zhang, B., and Horvath, S. 2005. A general framework for weighted gene co-expression network analysis. *Statist. Appl. Genetics & Mol. Biol.*, **4**(1).
- Zheng, Z., Lauritzen, J. S., Perlman, E., Robinson, C. G., Nichols, M., et al. 2018. A complete electron microscopy volume of the brain of adult *Drosophila melanogaster*. *Cell*, **174**(3), 730–743.
- Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., and Sun, M. 2020. Graph neural networks: A review of methods and applications. *AI Open*, **1**(Jan.), 57–81.
- Zhu, B., and Stölke, J. 2018. Subti Wiki in 2018: from genes and proteins to functional network annotation of the model organism *Bacillus subtilis*. *Nucleic Acids Res.*, **46**(D1), D743–D748.

Index

- Abel, Niels Henrik , 41
ablation , 462
activator , 140
ADHD , 501
adjacency matrix , 27, 39, 246, 399
 directed graph , 27
adjacency matrix, convention for , 141
admissible
 equation , 31, 448
 ODE , 31, 33
 system , 26
admissible graph , 177
Aharonov–Bohm effect , 218
algebraic geometry , 190
algorithm , 232, 236
 broken fibration symmetry , 368
Allen Brain Atlas , 467
Alon, Uri , 273
Alzheimer’s disease , 501
angular Gyrus , 477
anonymity , 56
anticorrelation , 399
AR loop , 286, 312
arabinose , 424
arc , 26
Argentina , 213
ArrayExpress , 404
arrow , 26
arrow type , 185
automorphism , 41–43, 206, 232, 233, 482
automorphism, vulnerability of , 203
autoregulation , 51
autoregulation loop , 286
AVAL , 453
AVBR , 453
axonal fiber , 447
axonal tract , 470
B. subtilis , 405
Babai’s algorithm , 65
bacterial genome , 288
Barabasi–Albert model , 437
base , 83, 86, 191
 minimum , 88
 space , 196
basin of attraction , 126
Bernal, John Desmond (J.D.) , 6, 7
Betti number , 390
bi-fan motif , 339
bifurcation , 139, 153, 154
 first , 155
 Hopf , 139, 154
 Hopf, symmetry breaking , 158
 local , 154
 pitchfork , 154
 saddle-node , 154
 steady-state , 139, 154
 transcritical , 154
bijection , 82
binary tree fiber , 312
binding site , 427
binomial distribution , 336
biology, axiomatization of , 206, 207, 209
biology, geometrization of , 210
biology, robustness of , 53
bipartite , 140, 142
bistability , 354
bistable , 364
Bloch’s theorem , 3
Bloch, Felix , 3
blood-oxygen-level-dependent signal , 476
BOLD , 476
BOLD time series , 477
Boolean kinetics , 268
brain , 397
 human , 476
brain clarification , 468
brain clarification , 467
brain injury , 501
branching ratio , 276, 282, 288, 314, 324, 327, 330,
 331
 high , 328
Brenner, Sidney , 392
broadcast , 55
Broca’s area , 477, 479
building block , 266, 273, 275, 319, 331, 449, 466
 classification , 281
 complex , 278
 complex fibration , 322
 composite Fibonacci , 324
 composite feed-forward Fibonacci , 327
 evolutionary dynamics , 312
fiber , 270

- code , 503
- dataset , 503
- fibration , 278, 333, 342
- for many species , 317
- intuition behind , 277
- simple , 278
- bundle
 - principal , 192
 - tangent , 192
 - trivial , 191
 - vector , 192
- c-Fos , 466, 467, 469
- C. elegans* , 47, 449
- C. elegans* connectome , 444, 445, 449
- Cardon–Crochemore algorithm , 235, 239, 241
- Cartesian product , 191
 - twisted , 191
- category theory , 190
- cell , 26
- cellular automaton , 54, 55
- Central Limit Theorem , 335, 336
- centrality analysis , 275
- chaos , 161
- circuit , 353
 - electronic , 355
 - gene regulatory , 355
- ClearMap , 467
- clique , 459
- clique synchronization , 459
- clock , 60, 303, 388
- clocked SR flip-flop , 365
- cluster , 35, 87
- cluster synchronization , 164
 - code , 504
- cluster synchronization manifold , 36, 165
- clustering , 275, 397, 404
- coarser , 104
- codon , 430
- coexpression , 275, 394, 398, 428, 432
- coexpression matrix , 436
- coin tossing , 336
- collective influence , 275
- Colombos , 404
- color refinement algorithm , 238
- color-isomorphic , 69, 71
- coloring
 - canonical , 241
- coloring , 35, 71
 - balanced , 36, 47, 71, 480
 - balanced, of Frucht graph , 108
 - minimal balanced , 442
- coloring subgroupoid , 122
- common neighbors similarity metric , 437
- community detection
 - algorithm , 275
- community detection , 333, 403
- commutator , 43
- compartment , 26
- complexity , 314
- complexity reduction , 373
- composite feed-forward Fibonacci , 324
- composite Feedback Fibonacci , 324
- compression , 488
- confusion matrix , 438
- connection , 195, 211
 - genetic , 225
 - in finance , 213
 - in physics , 212
- connectome , 6, 32, 435, 444, 465, 471
 - inferring from synchronization , 432
 - mouse , 470
 - repair , 454
 - symmetry of , 439, 480
- consensus matrix , 455
- conservation law , 30
- coreness index , 381
- CoReSym , 374
- correlation , 399, 467, 470, 476
- correlation matrix , 459
- correlation coefficient
 - Pearson , 397–399, 405, 407, 408
- correlation coefficient , 457
- correlation matrix , 398, 399, 407, 436
- covariogram , 399
- Crick, Francis , 222
- curvature , 193, 210, 218, 219, 226
 - genetic fibration , 220
 - in finance , 213
 - in physics , 212
- cycle , 226, 329–331, 388
- degradation , 264
- delay , 60
- dendrogram , 346, 404
- determinism , 55
- discretization , 181
- disease , 499
 - pathway , 500, 501
- DNA sequence , 222
- dynamics , 342
 - continuous , 60
 - discrete , 60
- dysregulation , 501
- E. coli* , 228, 288, 320, 376
- E. coli* TRN , 266, 281, 288
- Economics , 404, 405
- edge , 26, 27
- eigenmode , 163
- eigenvalue , 137
 - critical , 154
 - synchronous , 137
 - transverse , 137, 163
- eigenvector centrality , 275

- eigenworm , 449
 Einstein's equivalence principle , 3
 Einstein, Albert , 3, 220
 electromagnetism , 192, 213, 218, 219
 electroweak interaction , 219
 engram , 465, 466
 ε -pseudosymmetry , 246
 equitable partition , 235
 coarsest , 235, 237, 238
 equitable partition, algorithm , 241
 Erlangen Program , 209, 486, 487
 evolution , 5, 202, 445
 evolvability , 53
 expressive power of GNN , 488
 F-measure , 438
 FB-GradDesc , 495
 feed-forward fiber , 51, 138, 290
 feed-forward loop , 291, 338
 Feynman path integral , 228
 FFF , 51, 84, 201, 290, 388, 405
 FFL , 291, 338
 fiber , 47, 86, 191, 266, 268, 275, 332
 fiber bundle , 190, 196
 financial , 215
 over manifold , 190
 fiber number , 282, 316
 fiber, contrast with motif , 275
 Fibonacci circuit
 base , 101
 replica , 101
 Fibonacci fiber , 138, 309, 314, 324, 329
 Fibonacci number , 327
 fibration , 41, 47, 50, 83, 86, 102, 190, 206, 232, 248,
 263, 275, 394, 421, 468
 algorithms, 503
 application for essential brain areas , 485
 application in neurosurgery , 483
 building block , 278
 Calabi–Yau , 193
 formalism , 415
 graph , 190, 201
 Grothendieck , 199
 history of , 198
 Hopf , 195
 Hurewicz , 197
 in algebraic geometry , 195
 in category theory , 195
 in homotopy theory , 196
 injective , 379
 minimal , 88
 reconstruction of connectome , 475
 Serre , 197
 symmetry , 8, 47, 53, 80, 207, 487
 theory of brain , 483
 fibration-gradient descent , 495
 finer , 104
 finite-state automaton , 239
 finite-state machine , 54, 55
 FitzHugh–Nagumo equation , 30
 flip-flop , 231, 364
 Floquet theory , 130
 fluorophore , 466
 fMRI , 476
 foreign exchange , 213
 forward locomotion gap-junction circuit , 450
 fractal dimension , 276, 314, 324
 fragile , 77
 fragility , 57
 Frucht, Robert , 73
 function-structure relation , 432
 functional brain network
 percolation
 code , 506
 functional brain network influencer code , 506
 functional magnetic resonance imaging , 476
 functional network
 graphical lasso , 402
 Galois, Évariste , 41
 gauge invariance , 3, 213, 219
 gauge symmetry , 192
 in quantum mechanics , 192
 Gaussian distribution , 336
 Gell-Mann, Murray , 420
 gene duplication , 202, 224, 358, 359
 gene expression , 263, 413
 Gene Expression Omnibus , 404
 gene knockout experiment , 499
 gene regulatory code , 430
 general model , 140
 generalized diffusive coupling , 162, 164
 genetic code , 430
 Geometric Deep Learning , 486, 487
 Girvan–Newman algorithm, 346
 Glasso , 403
 golden ratio , 314
 Goodwin circuit , 133
 gradient descent , 495
 graph
 modification , 171
 weighted , 250
 typed , 27
 graph , 26, 29, 39
 biological , 26
 colored , 35, 185
 covering , 97
 directed , 26
 Erdős–Rényi , 40
 Frucht , 73
 symmetry groupoid , 107
 homomorphism , 81
 regular , 75
 simple , 29

- undirected , 27
- unweighted , 27
- weighted , 27, 181
- Graph Automorphism Problem , 44
- graph bundle , 198
 - Cartesian , 198
- graph fibration , 86, 89
 - quasi-fibration
 - code , 505
- graph isomorphism , 63
- Graph Isomorphism Problem , 65
- graph isomorphism test , 234
- graph product , 197
 - Cartesian , 197
 - direct , 197
 - lexicographic , 198
 - strong , 198
- graphical lasso , 402
- GRN , 139
- Grothendieck, Alexander , 41, 190, 199
- group , 3, 94
 - Abelian , 206
 - symmetric , 43
- group symmetry , 43
- groupoid , 41, 108, 117, 118, 120, 206, 207, 421
 - coloring symmetry , 113
- Haeckel, Ernst , 4
- head , 26
- heat map , 395
- Hebb, Donald , 447, 464
- Hebbian learning , 447
- hierarchical clustering algorithm , 455
- Hill function , 30, 139, 414, 424, 427
- Hodgkin–Huxley equation , 30
- Hodgkin–Huxley model , 448
- homomorphism , 201
- homotopy lifting property , 196
- homotopy theory , 196
- Hopcroft, John , 239
- hub , 275, 349
- hyperarc , 173
- hyperbolic , 132
- hyperbolicity , 171, 421
- hyperedge , 173, 179
- hypergraph , 29, 172, 173, 177
 - directed , 173
 - factor graph , 174
 - fibration , 175
 - hyperedge , 173
 - undirected , 173
- identity permutation , 44
- IEG , 466, 469
- immediate-early gene , 466
- in-degree , 57, 381
- information flow , 222, 223
- information vortex , 390
- injective , 81
- input
 - isomorphic , 32
 - isomorphism, 32
- input , 27
 - isomorphic , 111
 - isomorphism , 111
- input set , 14, 57, 58, 112
- input set color vector , 254
- input tree , 50, 58, 112, 201, 268, 279, 288, 314, 324
 - as set of paths , 61
 - isomorphism , 64
- integration vs. segregation problem , 403
- interaction
 - many-body , 29
 - two-body , 29
- interneuron , 455
- ISCV , 254
- Jaccard similarity , 437
- Jacobian matrix , 132
- JK flip-flop , 366
- Johnson's algorithm , 390
- Jordan, Camille , 41
- k -core , 275, 381
- k -core decomposition , 381
- k -shell , 381
- Katz index , 437
- Klein, Felix , 209, 487
- Kronecker product , 163
- Kupka–Smale Theorem , 132
- Kuramoto model , 127
- L^1 norm , 245
- Lac operon , 263
- Lagrange, Joseph-Louis , 41
- Landau, Lev , 3
- language , 476
- Laplacian , 161, 162, 164
- lattice , 103, 216
 - gauge theory , 216
- level of synchrony measure , 458
- lifting , 89
- lifting property , 88–90
- link , 26
 - prediction , 435, 436, 439
- lock-on , 138, 144, 156, 179
- locomotion , 448
- logarithmic spiral , 194
- logic circuit , 388
- loop , 29
- Lorentz invariance , 3
- Lorentz, Hendrik , 3
- Louvain algorithm , 346, 347
- machine learning , 404, 486
- magnetic flux , 218
- Maldacena, Juan , 213

- mass balance , 30
 master stability equation , 163
 master stability function , 127, 161, 163
 for clusters , 164
 maximum entropy , 399
 maximum Liapunov exponent , 163, 167
 Maxwell's equations , 218
 McKay, Brendon , 235
 memory , 388, 465, 466
 encoding , 468
 retrieval , 468
 updating , 468
 mesoscale connectome , 447, 475
 metabolator , 50, 83, 115, 138, 151
 metabolic pathway , 264
 metabolome , 32
 Michaelis–Menten kinetics , 30
 microarray , 398
 MILP algorithm , 455, 481
 minimal TRN , 383
 minimum op-fibration base , 95
 Minkowski spacetime , 192
 Missing link
 algorithm , 436
 missing link , 433, 438, 443
 Möbius band , 191, 194
 mode , 161
 model equation, construction of , 30
 modularity , 275, 333, 345, 403
 Louvain , 346
 module , 333
 Monod, Jacques , 6, 7
 morphism , 120
 morphogen , 5
 Morris–Lecar equation , 30
 motif , 273, 315, 332, 333
 binding , 427
 calculation of , 334
 drawbacks of , 341
 motorneuron , 455
 mouse brain , 467
 mRNA , 264
 multi-Layer perceptron , 492
 multiedge , 29
 multigraph , 29
 homomorphism , 81
 multiple edge , 29
 multiset , 239
 multivariate Gaussian distribution , 402
 mutation , 203
 mutual information , 399
n-ary tree fiber , 312
 Nauty algorithm , 44, 65, 235, 236, 504
 network
 carbon utilization , 407
 network , 26, 39
 Bayesian , 399
 biological , 26
 convolutional neural , 487
 dynamical system , 32
 enzyme , 319, 320, 328
 functional , 397, 408
 gene regulatory , 6, 139
 glycolysis , 328
 graph neural , 487
 groupoid , 121
 homogeneous , 104
 influence , 30, 94, 416
 information-processing , 379
 lumped , 264
 metabolic , 176, 184, 319
 molecular reaction , 30
 multi-layered , 184
 neuronal , 468
 oxidative enzyme , 328
 scale-free , 275, 437
 small-world , 275
 social , 437
 structural , 397
 transcriptional regulatory , 139, 263
 weighted , 39, 181
 network
 functional , 459
 protein-mRNA , 139
 network reconstruction
 code , 505
 neural influencer code , 506
 neuromodulator , 448
 neuron
 activated , 466
 Newton, Isaac , 220
 node , 26
 with no inputs , 187
 node type , 185
 Noether's theorem , 3
 Noether, Emmy , 3
 noise , 250, 251
 noise matrix , 249
 Norris's theorem , 67, 236
 NP-complete , 65
 NP-hard , 233
 null hypothesis , 336
 object , 120
 objective function , 251
 op-fibration , 95
 operon , 263, 270, 315, 332
 optimization , 251
 optogenetics , 466
 orbit , 45
 orchestra , 38
 out-degree , 381
 output , 27

- output trees , 95
 p -value , 335
 Pagerank , 275
 parallel transport , 210, 215, 225
 parameter changes , 171
 partial derivative, notation for , 133
 partially ordered set , 103
 partition , 26, 35
 - equitable , 47, 67, 71, 87
 - equitable, examples , 72
 - minimal balanced coloring , 71
 - minimal equitable , 50, 68, 88
 - minimal orbital , 47
 - orbital , 44, 45
 path , 28, 59
 pattern of connectivity , 333
 PBCIP , 250, 251
 penalty parameter , 403
 percolation analysis , 402
 permutation , 42
 - matrix , 42, 246
 Perron–Frobenius Theorem , 276, 327
 perturbation , 137
 - matrix , 249
 - theory , 420, 421
 phase locking value , 399, 477
 phase transition , 3
 phenotype , 205
 phonemic fluency , 476
 phyllotaxis , 4
 Poincaré map , 130
 Poincaré section , 130
 polydiagonal subspace , 36
 Practical Graph Isomorphism , 65
 preferential attachment index , 437
 principal component , 275
Principia Mathematica , 220
 PRN , 139, 142
 probability distribution , 336
 protein , 264
 protein-protein interaction , 499
 proteome , 32
 pseudo-balanced coloring , 250
 - code , 505
 - integer program , 250
 pseudofibration , 421
 pseudosymmetry , 245–247, 421, 446
 quantum chromodynamics , 193
 quasi-equitable , 249
 quasifibration , 249, 446
 quiver , 199
 quotient
 - graph , 47, 48
 - network , 49, 87, 202
 radiolaria , 4
 Randic index , 437
 reaction-diffusion equation , 76
 reconstruction algorithm , 436
 recurrence relation , 327, 329
 refine , 104
 region of interest , 474
 regulon , 288, 315, 394
 RegulonDB , 320, 376
 renormalization group , 374
 repressilator , 138, 158, 353
 repressor , 140
 rich-club , 275
 rigid , 44, 171
 Rigid Equilibrium Theorem , 78, 414
 ring oscillator , 356
 RNA-seq , 398
 robust , 77
 - structural , 171
 - technical meaning , 171
 - informal usage , 202
 ROI , 467, 474
 Ruffini, Paolo , 41
 Salton index , 437
 SAT-FFF , 296
 satisfied feed-forward fiber , 296
 scale-free metric , 437
 SCC , 28, 228, 268, 320, 330, 382, 388
 SCC of *E. coli* , 268
 schizophrenia , 501
 sector , 205
 Serre, Jean-Pierre , 190
 Shoemaker, Ken , 196
 signal propagation , 264
 simple Fibonacci fiber , 309
 simple multilayer fiber , 306
 simplicial homology , 390
 single cell resolution , 452
 Smolen circuit , 83, 138, 151
 Smolen oscillator , 50, 116, 354, 388
 software implementation , 254, 503
 source , 26
 special model , 140
 stability , 45, 55, 126, 170
 - asymptotic , 130
 - exponential , 129
 - Liapunov , 130
 - linear , 129, 133
 standard deviation , 164
 Standard Model , 3, 206, 374, 420
 state , 32, 55
 statistical abundance , 333
 string theory , 193
 stroke , 501
 strongly connected , 28, 268
 strongly connected component , 28, 228, 266, 268, 382
 structural stability , 171
 structure-function relation , 26, 40, 54, 205, 394

- subgraph , 28
 induced , 28
 Subgraph Isomorphism Problem , 65
 subgroupoid , 118
 SubtiWiki , 404
 surjective , 82
 symmetry , 44, 397
 approximate , 245
 fibration , 80
 group , 41
 groupoid , 107, 110, 114
 isomorphism , 113
 necessity for , 422
 selection pressure for , 428
 variations from , 420
 symmetry breaking , 53, 312, 418, 422, 423
 caused by delays , 60
 forced , 420
 in human language connectome , 482
 in physics , 362
 spontaneous , 420
 synaptic connection , 466
 synchronizability , 45, 56, 126
 synchronizability index , 164
 synchronizable, completely , 36
 synchronization , 71, 466, 467, 470
 cluster , 8, 26, 36, 38, 44, 394, 432, 477, 480
 complete , 26, 36, 39
 imperfect , 398
 perfect , 423
 synchronous update , 59
 synchrony , 466
 cluster , 125
 measure of , 457
 necessary conditions , 56
 perfect , 413
 quasi-perfect , 413
 subspace , 36, 57
 synchrony-breaking , 53, 137, 163
 synchrony-preserving , 137, 163
 synthetic biology , 353
 systems biology , 273
 tail , 26
 target , 26
 TF , 264
 Thompson, D'Arcy , 4, 5, 7
 threshold, choice of , 400
 thresholding , 181, 399, 403, 404, 407, 436, 479
 toggle-switch , 138, 147, 230, 354, 364, 384, 386, 388
 transcription factor , 223, 264
 transcriptional unit , 270
 transcriptome , 32, 395, 404
 transistor , 356
 transverse Jacobian , 137
 TRN , 139, 263
 as a computer , 367
 TU , 270
 Turing machine , 392
 Turing pattern , 5
 Turing, Alan , 5, 7
 type, of edge , 27
 type, of node , 27
 unbalanced coloring, conditions for , 78
 Unger, Stephen , 238
 uniformity assumption , 415, 416, 418, 427
 unit , 26
 UNSAT-FFF , 299, 358
 unsatisfied feed-forward fiber , 299
 upstream subnetwork , 276
 valence , 57
 variability , 433
 Varshney connectome , 460
 verb generation , 476
 vertex , 26, 27
 vertex symmetry , 179
 Vigna, Sebastiano , 199
 von Neumann cellular automaton , 392
 von Neumann machine , 392
 weight , 181
 Weighted Gene Correlation Network Analysis , 436
 Weisfeiler–Lehman test , 235, 488, 489
 Wernicke's area , 477, 479
 Weyl, Hermann , 213, 219
 WGCNA , 436
 white matter tract , 447, 474–476, 480, 481
 Widlar current-mirror circuit , 358
 WL-test , 235, 488
 Z-score , 335, 336, 343
 Zeeman effect , 363