Linear Systems on Graphs

Controllability of Structured Networks

Graduate Lecture Notes

Nam-Jin Park
Major of Control and Instrumentation Engineering, Pukyong National University

Fall 2026 • Version 1.9 (draft), 2026-09-05

Major of Control and Instrumentation Engineering, Pukyong National University, Busan, Republic of Korea.

Suggested citation: N.-J. Park, *Linear Systems on Graphs: Controllability of Structured Networks*, Version 1.9 (draft), 2026-09-05. Graduate lecture notes, Pukyong National University.

The containment figure (Fig. 6.2) and the three orientation panels (Fig. 8.2) are reproduced from the author's doctoral dissertation [1] and are the author's own material.

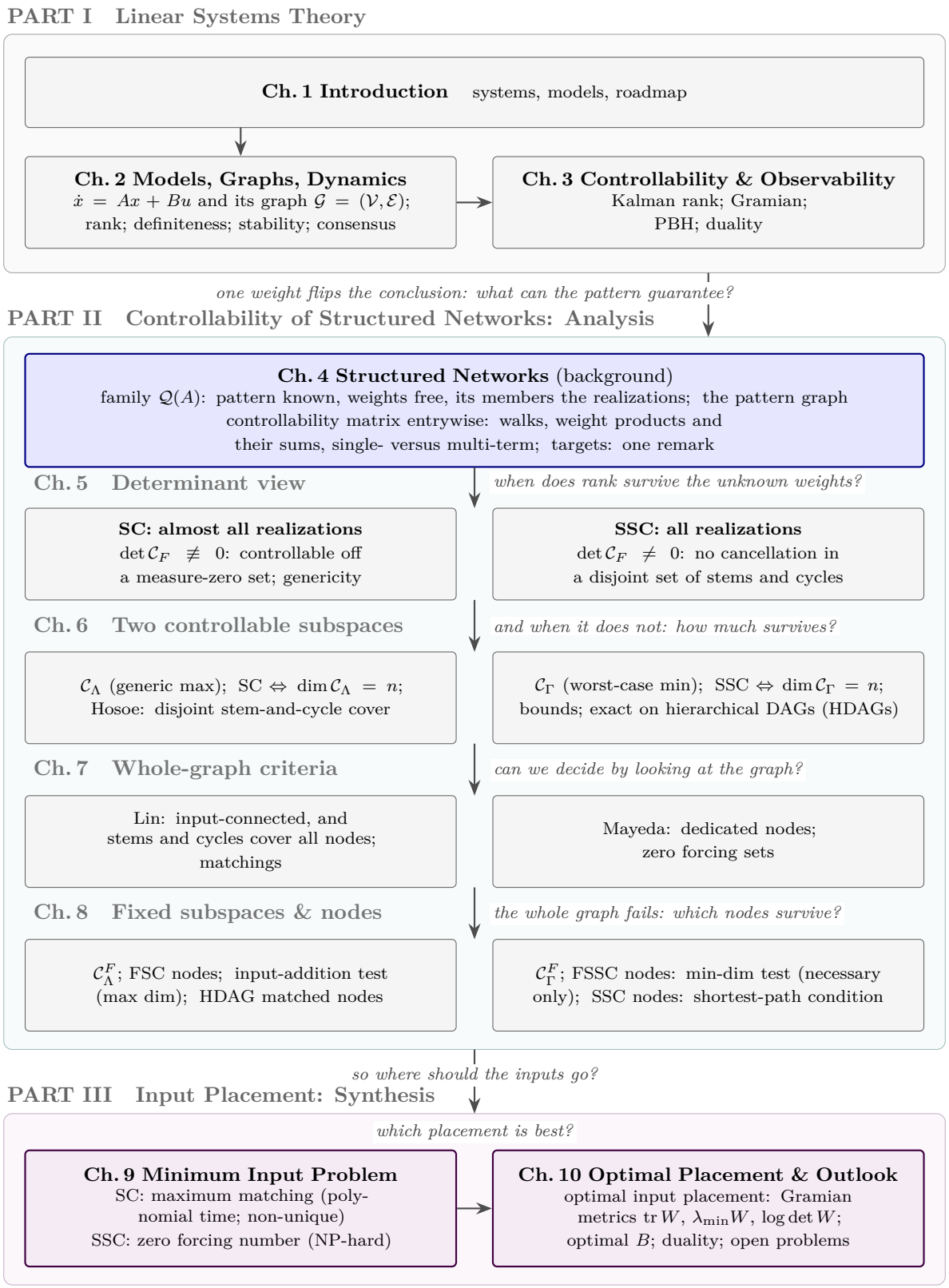
Preface

These notes serve a one-semester graduate course on linear systems, with a second half on networks. Part I is the classical theory (the state-space model $\dot{x} = Ax + Bu$, $y = Cx$, its solutions and its stability) and the two questions that organize everything after it: can the input steer the state anywhere, and does the output reveal the state? The scope follows Chen [2].

Alongside the matrices we carry a picture. Draw one node for each state variable and one for each input and each output, and an arrow from j to i whenever j influences i, that is, whenever the corresponding entry of A, B or C is non-zero. A linear system is then a weighted directed graph, and the two questions become questions about that graph: which nodes does an input reach, and how does its effect spread?

Part II asks what survives when the numbers on the arrows are unknown. In a power grid, a robot team or a gene network, who influences whom is reliable knowledge, but how strongly is usually not. Much of the theory survives anyway (controllability can often be decided, or shown to fail, from the arrows alone), and the arguments become combinatorial, organized around one polynomial, the determinant of the controllability matrix. Part III turns analysis into synthesis and asks where the inputs should be attached: how few suffice, and which placement steers the network with least effort. The figure on the following page shows how the book is organized, and the Notation page after the contents collects the symbols that recur across chapters.

Prerequisites are an undergraduate course in linear systems and a working command of linear algebra. Readers who already know the material of Chen [2] may take Chapter 2 and Chapter 3 at review pace and begin in earnest at Chapter 4. Its graph-theoretic background, Section 4.3, is written as a reference to be consulted on demand rather than read straight through, and Appendices A and B collect the longer proofs, which may be left for a second reading. Exercises close every chapter, and Section 10.4 lists the open problems offered as semester projects.



Roadmap of the book. Part I develops classical linear system theory up to the Kalman rank condition. One perturbed entry flipping the controllability conclusion motivates Part II, where only the zero/non-zero pattern is reliable. Four analysis levels follow, each split into a cell for structural controllability (SC) and one for strong structural controllability (SSC), that is, almost all versus all realizations. Every condition appears in both algebraic and graph-theoretical form, and the levels are linked by the questions each one leaves open. Part III turns analysis into synthesis: the minimum input problem, then Gramian-optimal placement beyond the minimum.

Contents

List of Figures

Notation

Symbols used across chapters. A symbol not listed here is local to the proof or example that introduces it.

Symbol	Meaning	Defined at
$\mathcal{G} = (\mathcal{V}, \mathcal{E})$	system graph	Definition 2.3
$\mathcal{V}^S, \mathcal{V}^I, \mathcal{V}^O$	state, input and output nodes	Definition 2.3
$(j, i) \in \mathcal{E}$	edge, present iff $[A]_{ij} \neq 0$	Definition 2.3
e_k	the k-th standard basis vector of $\mathbb{R}^n$	this page
$\mathcal{Q}(A)$	family of a pattern matrix; its members are realizations	Definition 4.1
$\mathcal{C}$, $\mathcal{O}$	controllability and observability matrices; $\mathcal{C}_F$ at one realization	Theorems 3.2 and 3.11
W	controllability Gramian[b]	Theorem 3.4
V, Λ	eigenvector matrix and eigenvalue diagonal	Section 2.5
$\mathcal{L}$, D	graph Laplacian and degree matrix[b]	Section 2.9
m	number of inputs; written p in Section 8.5 and Appendix B	Eq. (2.1)
n, q	state and output dimensions	Eq. (2.1)
WP, SWP	weight product, sum of weight products	Section 4.4.3

Symbol	Meaning	Defined at
$\mathcal{C}_\Lambda$, $\mathcal{C}_\Gamma$	structurally / strongly structurally controllable subspace[a]	Definition 6.1
$\mathcal{C}_\Lambda^F$, $\mathcal{C}_\Gamma^F$	fixed subspaces: $\operatorname{Im}\mathcal{C}_F$ intersected over the generic-dimension realizations, resp. over all realizations	Definition 8.2
ℓ_k	distance layers	Definition 4.21
$R(S)$	reachable set	Definition 4.12
$N^{\text{in}}, N^{\text{out}}, N$	in-, out- and undirected neighbors	Definition 4.5
S	the subset of state nodes under test	Definition 7.7
b, B	input matrix, written b for its single column where the pattern has exactly one input	Eq. (2.1)
z	left null vector of the PBH test[c], the left eigenvector the input cannot excite	Theorem 3.8
$\mathcal{D}$	driver set	Problem 9.1
ν	size of a maximum matching on the state subgraph	Theorem 9.1
$Z(\mathcal{G})$	zero forcing number[c]	Definition 9.5
$q^\star$, $m^\star$	shortest-stem counts	Theorem B.1
$\mathcal{V}^u_{\text{int}}$	intersection of the control paths from u	Theorem 8.17
$\bar{\mathcal{G}}$	the edge-deleted subgraph	Section B.7

[a] As a subscript of $\mathcal{C}$, Λ marks the generic maximum and Γ the worst-case minimum. Some

of the cited literature assigns the two letters the other way round. Standing alone, Λ is the eigenvalue diagonal of Section 2.5.

[b] Two letters carry a second, universal meaning: D is the degree matrix of Section 2.9 and the feedthrough matrix of Eq. (2.1). W is the Gramian of Theorem 3.4 and the weight matrix of Section 2.9. Context separates them and neither is renamed.

[c] $Z(\cdot)$ is the zero forcing number of a graph. The $Z(\pi)$ of Section A.1 is the zero set of a polynomial and is local to that proof. The letter $\mathcal{S}$ is local in the same way: the diagonal self-loop matrix of Remark 9.9 in one place, a node-disjoint set of stems inside a proof of Appendix A in the other. So is the z of the linear system $Mz = w$ in Section 2.4, which is the unknown of that one sentence and not the PBH vector above.

Part I

Linear Systems Theory

Chapter 1

Introduction: Systems, Models, and the Plan of the Book

Why we study models rather than physical systems, what an analytical study of a system consists of, and where this book is headed: from a single plant with known parameters to networks whose interconnection pattern is the only reliable knowledge.

1.1 Physical Systems and Models

Engineering deals with physical systems (motors, vehicles, circuits, fleets of robots), but engineering *analysis* deals with models. The distinction is worth keeping sharp. A resistor with a fixed resistance is a model: the physical component varies with temperature and burns out beyond its rating. A rigid link of a manipulator is a model: the physical link flexes. Whenever we compute, we compute on the model, and the value of the computation is bounded by the quality of the model.

Which model is appropriate depends on the question being asked. A quadrotor is a point mass to a path planner, a rigid body to an attitude controller, and a set of coupled motor–propeller dynamics to a vibration analyst. These three models of one aircraft do not contradict one another. Throughout this book the word *system* means a model of a physical system, and we assume the modeling step has been done: our starting point is the model itself.

An analytical study proceeds in four stages: modeling, mathematical description, analysis, and design. Analysis splits further into *quantitative* questions (what is the response to this input?) and *qualitative* ones: is the system stable? can the input steer every state? does the output reveal the state? The qualitative questions are where design insight comes from, and they run through this book: from Chapter 3 onward, almost every page is about one qualitative property, controllability, asked under progressively weaker knowledge of the model.

1.2 From One Plant to Networks

Classical control theory grew around a single plant whose parameters are known (measured on a test bench, identified from data, printed on a datasheet). The state-space theory of Part I assumes that setting: a matrix A full of reliable numbers.

The systems that dominate modern applications are different. A power grid, a formation of mobile robots, a gene regulatory network, a social network: each is a collection of simple

units coupled by an interconnection structure. For such systems the roles of knowledge are reversed. The *structure* (who is coupled to whom) is usually reliable: it is the wiring diagram, the communication topology, the regulatory map. The *edge weights* are usually not: they vary, they depend on operating point, they are expensive or impossible to measure at scale, and in a robot team they may be redesigned tomorrow.

This inversion is the premise of Part II. If the numbers in A cannot be relied on but the zero/non-zero pattern can, what remains of controllability? Remarkably much, and the answers are combinatorial: they are read off a graph. For that reason this book puts the graph at the very beginning of the theory (Chapter 2 introduces every linear system together with its graph), so that when the numbers are finally dropped in Chapter 4, the picture that remains is already familiar.

1.3 Book Outline

The three parts run as follows. Part I develops the classical state-space theory: the model together with its graph (Chapter 2), then controllability and observability up to the Kalman, Gramian and PBH tests (Chapter 3). Part II opens with vocabulary-building background (Chapter 4) and then proceeds through four *analysis* levels: definitions and the determinant view (Chapter 5), controllable subspaces (Chapter 6), whole-graph criteria (Chapter 7), and node-level criteria (Chapter 8). Part III turns analysis into *synthesis*: the minimum input problem (Chapter 9) and optimal placement beyond the minimum (Chapter 10). Two axes run through every level: structural versus strong structural (almost all versus all realizations of the family, Chapter 4), and algebra versus graph (every condition stated in both forms). The roadmap figure in the Preface shows this organization on one page. Two appendices follow Part III and collect the longer proofs: Appendix A on the SC side and Appendix B on the SSC side. Most sections restate a result of the main text by cross-reference and prove it. A few state in full what the main text records only in prose, among them the shortest-stem lower bound and the exact fixed-node criterion.

Chapter 2 reviews, at graduate pace, exactly the material this book uses. Worked examples are small by design (most fit on four nodes and can be verified by hand), and each concept is illustrated on the smallest graph that exhibits it.

Sources. Part I follows the scope of [2], and the networks perspective draws on [3, 4].

Exercises

Exercise 1.1 (The four stages)**.** Name the four stages of an analytical study of a system, in the order given in this chapter. State which stage the book assumes has already been carried out, and which stage occupies almost every page from Chapter 3 onward.

Exercise 1.2 (Quantitative and qualitative)**.** Classify each question as quantitative or qualitative: (i) what is the output two seconds after a unit step is applied? (ii) is the system stable? (iii) can the input steer the state to any target? (iv) does the output determine the state? Which two of them become the subject of Chapter 3?

Exercise 1.3 (Structure or weights)**.** Decide true or false, with one sentence of reason: *for a power grid or a formation of mobile robots, the individual edge weights are known more reliably than the interconnection structure.*

Exercise 1.4 (The two axes)**.** Two axes run through every analysis level of Part II. Name both, and state the two alternatives on each.

Chapter 2

Foundations: Models, Graphs, and Dynamics

Everything the book needs from classical theory, in one chapter at review pace: the state-space model and (immediately) its graph, the linear-algebra background (rank, eigenstructure, definiteness), solutions and stability, and a closing application on a network: Laplacian dynamics and consensus. From here on, a linear system is *a weighted directed graph.*

2.1 State-Space Models

Throughout the book a system is linear, time-invariant, and lumped: its memory is a finite-dimensional vector $x(t) \in \mathbb{R}^n$, the *state*, evolving as

$$\dot{x} = Ax + Bu, \qquad y = Cx + Du, \tag{2.1}$$

with input $u \in \mathbb{R}^m$, output $y \in \mathbb{R}^q$, and constant matrices $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$, $C \in \mathbb{R}^{q\times n}$, $D \in \mathbb{R}^{q\times m}$ (almost always $D = 0$ here). $x(t_0)$ together with $u(t)$ for $t \geq t_0$ determines the entire future.

Example 2.1 (Mass–spring–damper)**.** A mass m on a spring k with damping c and force input u obeys $m\ddot{q} = -kq - c\dot{q} + u$. With $x = (q, \dot{q})$,

$$A = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{c}{m} \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

if the position is measured.

Example 2.2 (DC motor)**.** Armature current i and shaft speed ω with voltage input: $L\dot{i} = -Ri - k_e\omega + v$, $J\dot{\omega} = k_t i - b\omega$. With $x = (i, \omega)$,

$$A = \begin{bmatrix} -\frac{R}{L} & -\frac{k_e}{L} \\ \frac{k_t}{J} & -\frac{b}{J} \end{bmatrix}, \quad B = \begin{bmatrix} \frac{1}{L} \\ 0 \end{bmatrix}, \quad C = \begin{bmatrix} 0 & 1 \end{bmatrix}.$$

Both examples are written in the letters of their own domains: m, k, c, q for mass, stiffness, damping and position, and L, R, J, b, i for inductance, resistance, inertia, viscous friction and

current. None of these letters carries here the meaning it carries elsewhere in the book, where m and q are the input and output dimensions of Eq. (2.1), b and c are the single input column and the single output row from Example 2.4 on, i and k are node and step indices, $R(S)$ is the reachable set of Definition 4.12, and the graph Laplacian of Section 2.9 is $\mathcal{L}$, never L.

Two remarks close the section. Sampling Eq. (2.1) with period T gives a discrete-time model $x[k{+}1] = A_d x[k] + B_d u[k]$ with $A_d = e^{AT}$ (Section 2.7). And eliminating the state by the Laplace transform gives the *transfer matrix* $\hat{G}(s) = C(sI - A)^{-1}B + D$, the external description. We use it sparingly.

2.2 Graph of Linear Systems

Here is the definition that sets the viewpoint of this book.

Definition 2.3 (Graph of a linear system)**.** The system Eq. (2.1) defines a weighted directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with node set $\mathcal{V} = \mathcal{V}^S \cup \mathcal{V}^I \cup \mathcal{V}^O$ (one node per state, input, and output) and edges

$$(j,i) \in \mathcal{E} \iff [A]_{ij} \neq 0, \qquad (u_k, i) \in \mathcal{E} \iff [B]_{ik} \neq 0, \qquad (i, y_k) \in \mathcal{E} \iff [C]_{ki} \neq 0,$$

each carrying the corresponding entry as its weight. State nodes that receive an input edge are *driver* (or *leader*) nodes.

Note the direction: $[A]_{ij} \neq 0$ means state j *influences* state i, an edge from j to i. Thus A acts as a weighted adjacency matrix (in the transpose convention, which we fix once and use everywhere). Every linear system can now be *drawn*.

Example 2.4 (A four-node network)**.** The system

$$A = \begin{bmatrix} 0 & 0 & -1 & 0 \\ 2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 3 & 1 & 0 \end{bmatrix}, \qquad b = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \qquad c = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}$$

has the graph of Fig. 2.1: a cycle $1 \to 2 \to 3 \to 1$ plus the edges $2 \to 4$ and $3 \to 4$, the input driving node 1 and the output reading node 4. Matrix and picture carry identical information, and the picture will often be the faster one to reason on. This system returns in Chapter 3.

$$A = \begin{bmatrix} 0 & 0 & -1 & 0 \\ 2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 3 & 1 & 0 \end{bmatrix}, \quad b = e_1, \; c = e_4^\top$$

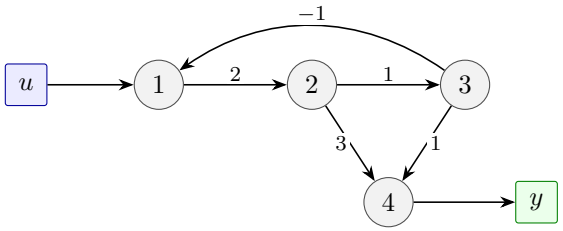


Figure 2.1: The four-node network of Example 2.4: the matrices (left) and their graph (right) are the same object. Input and output nodes are drawn as squares.

Part II will do exactly one thing to this picture: forget the numbers on the edges and keep only which edges exist.

2.3 Worked Models as Graphs

The graph view is not reserved for networks. Even a textbook plant is a graph. For the mass–spring–damper of Example 2.1, $[A]_{12} = 1$ gives an edge $\dot{q} \to q$, the entries $-k/m$ and $-c/m$ give an edge $q \to \dot{q}$ and a self-loop at $\dot{q}$, and the input drives $\dot{q}$ (Fig. 2.2, left).

For a *network* the graph is the natural starting point rather than something derived from the matrix: four robots exchanging relative information over communication links form the graph directly, and the matrix is read off the picture (Fig. 2.2, right). The dynamics that such information exchange generates (consensus) closes this chapter in Section 2.9.

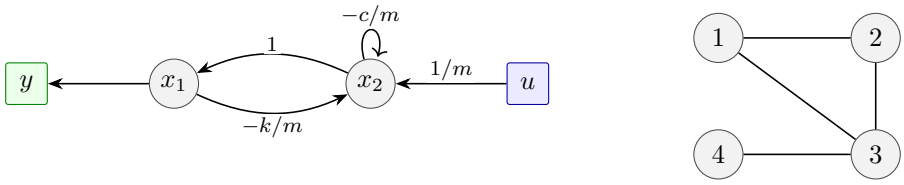


Figure 2.2: Left: the mass–spring–damper of Example 2.1 drawn as a graph. Right: a four-robot communication topology (undirected edges = bidirectional links). Its dynamics are built in Section 2.9.

2.4 Rank and Linear Equations

The tool this book uses most often is the rank of a matrix: the dimension of its column space $\operatorname{Im} M$, the set of all vectors Mz. The system $Mz = w$ is solvable iff $w \in \operatorname{Im} M$, and uniquely solvable iff in addition $\ker M = \{0\}$. The sum $\operatorname{rank} M + \dim \ker M$ equals the number of columns. Two points to retain from this review:

Rank is a spanning question. For

$$M = \begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 1 \\ 2 & 1 & 5 \end{bmatrix}$$

the third row is twice the first plus the second, so $\operatorname{rank} M = 2$: the columns span a plane. We will constantly ask "do these columns span $\mathbb{R}^n$?" Controllability is precisely such a question.

Rank never grows under multiplication: $\operatorname{rank}(MN) \le \min\{\operatorname{rank} M, \operatorname{rank} N\}$, and selecting rows or columns can only lower it. Both facts are used in every chapter.

2.5 Eigenstructure and Matrix Functions

If $Av_i = \lambda_i v_i$ with n independent eigenvectors, then $A = V\Lambda V^{-1}$ with $V = [v_1 \ \cdots \ v_n]$ and $\Lambda = \operatorname{diag}(\lambda_1, \dots, \lambda_n)$, and every polynomial or power series in A is computed on the eigenvalues. When eigenvectors are deficient the Jordan form plays the same role (we use it only at statement level).

Example 2.5. $A = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix}$ has $\lambda_1 = -1$, $\lambda_2 = -2$ with eigenvectors $v_1 = (1, -1)$, $v_2 = (1, -2)$. This A serves again in Sections 2.7 and 2.8.

Part II works mostly with the plain *powers* A^k. The key identity of the book makes those powers combinatorial objects: entries of A^k enumerate the walks of length k (edge sequences that may revisit nodes, Definition 4.7) on the graph of Definition 2.3 (Chapter 4).

2.6 Quadratic Forms and Definiteness

A symmetric $M = M^\top$ has real eigenvalues and an orthonormal eigenbasis, so the quadratic form $x^\top Mx$ is a weighted sum of squares along that eigenbasis. $M \succ 0$ (positive definite) iff all eigenvalues are positive, $M \succeq 0$ (positive semidefinite) iff none is negative. Leading principal minors give the classical test.

Example 2.6. Three cases: $\begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix} \succ 0$ (eigenvalues $1, 3$), $\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \succeq 0$ (eigenvalues $0, 2$, the zero eigenvector $(1, -1)$ giving a direction along which the form vanishes), and $\begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}$ indefinite (eigenvalues $3, -1$).

Quadratic forms are used three times later: for the graph Laplacian (Section 2.9), for the Lyapunov stability condition (Section 2.8), and for the Gramian-based placement metrics of Chapter 10.

2.7 Solutions of State Equations

The unique solution of Eq. (2.1) is

$$x(t) = e^{At}x(0) + \int_0^t e^{A(t-\tau)}Bu(\tau)\,d\tau, \tag{2.2}$$

a zero-input term carrying the initial state and a zero-state term accumulating the input along the flow. The matrix exponential is computed by series, by eigenstructure, or by taking the inverse Laplace transform of $(sI - A)^{-1}$.

Example 2.7. For A of Example 2.5, expanding the initial condition in the eigenbasis gives

$$e^{At} = \begin{bmatrix} 2e^{-t} - e^{-2t} & e^{-t} - e^{-2t} \\ -2e^{-t} + 2e^{-2t} & -e^{-t} + 2e^{-2t} \end{bmatrix};$$

both modes decay, as Section 2.8 confirms. Sampling with period T gives $A_d = e^{AT}$ exactly. Discretization is evaluation of the flow, not approximation.

2.8 Stability

The zero-input response $e^{At}x(0)$ decays for every $x(0)$ iff every eigenvalue of A has negative real part (A is *Hurwitz*). Marginal cases sit on the imaginary axis. The eigenvalue test has an algebraic counterpart that avoids computing the spectrum:

Theorem 2.8 (Lyapunov)**.** *A is Hurwitz iff for some (equivalently every) $Q \succ 0$ the Lyapunov equation $A^\top M + MA = -Q$ has a solution $M \succ 0$.*

Example 2.9. For A of Example 2.5 and $Q = I$, solving the three scalar equations of $A^\top M + MA = -I$ gives

$$M = \begin{bmatrix} \frac{5}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} \end{bmatrix} \succ 0 \qquad (\text{minors } \tfrac{5}{4} > 0,\ \det M = \tfrac{1}{4} > 0),$$

establishing stability without ever finding an eigenvalue. The quadratic form $x^\top M x$ decreases along every trajectory.

BIBO (input–output) stability is implied by, but strictly weaker than, the internal stability above. The gap is explained by controllability and observability in Chapter 3.

2.9 Laplacian Dynamics and Consensus

The chapter closes with all of its tools applied at once to the network of Fig. 2.2 (right). Let W be the symmetric adjacency matrix of an undirected graph ($w_{ij} = w_{ji} > 0$ on edges) and $D = \operatorname{diag}(W\mathbf{1})$ (the degree matrix, unrelated to the feedthrough D of Eq. (2.1)). The *Laplacian* is $\mathcal{L} = D - W$. Each agent moves toward its neighbors $N(i)$ (Definition 4.5, and on an undirected pattern the in- and out-neighbors coincide):

$$\dot{x}_i = \sum_{j \in N(i)} w_{ij}\,(x_j - x_i) \qquad \Longleftrightarrow \qquad \dot{x} = -\mathcal{L}x. \tag{2.3}$$

In the sense of Definition 2.3, the graph of $\dot{x} = -\mathcal{L}x$ is that undirected graph with each edge read as two opposite directed edges of weight w_{ij}, and with a self-loop of weight $-[D]_{ii}$ (minus the degree) at every node i.

Example 2.10 (Four agents)**.** For the topology of Fig. 2.2 (right) with unit weights (edges $12, 23, 34, 13$),

$$\mathcal{L} = \begin{bmatrix} 2 & -1 & -1 & 0 \\ -1 & 2 & -1 & 0 \\ -1 & -1 & 3 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix},$$

with row sums zero by construction. Its eigenvalues are $0, 1, 3, 4$.

Three spectral facts, each an application of an earlier section:

1. $\mathcal{L}\mathbf{1} = 0$: the all-ones vector is an eigenvector with eigenvalue 0. Thus agreement is an equilibrium.
2. $x^\top \mathcal{L} x = \frac{1}{2}\sum_{i,j} w_{ij}(x_i - x_j)^2 \geq 0$ (the factor $\frac{1}{2}$ because each undirected edge is counted in both orders), so $\mathcal{L} \succeq 0$ by Section 2.6: all remaining eigenvalues are positive or zero.
3. The graph is connected iff $\operatorname{rank}\mathcal{L} = n-1$, iff the second smallest eigenvalue λ_2 (the *algebraic connectivity*) is positive.

Consequently, on a connected graph every solution of Eq. (2.3) converges to average agreement, $x(t) \to (\frac{1}{n}\mathbf{1}^\top x(0))\,\mathbf{1}$ (*average consensus* in the cited literature), and the disagreement decays at rate λ_2, which is 1 for the four agents of Example 2.10. The quadratic form $x^\top \mathcal{L} x$ itself is the Lyapunov function establishing it. Figure 2.3 integrates both statements on those four agents, and compares the resulting rate with the rate the same four agents achieve once the chord 1–3 is removed. For a directed graph with a rooted spanning tree, consensus is still

reached, but to a *weighted* agreement value (the left-Perron-vector combination of the initial states). The simple average is recovered only for balanced directed graphs (stated without proof, see the references).

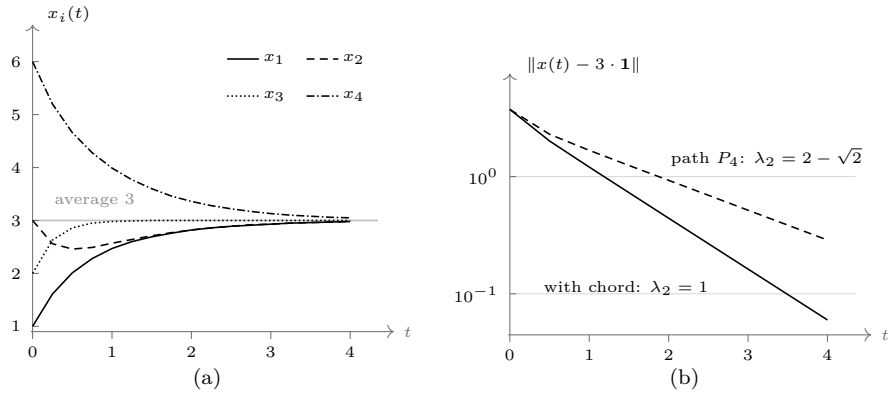


Figure 2.3: Consensus on the four agents of Example 2.10, integrated from $x(0) = (1, 3, 2, 6)^\top$. (a) The four states meet at the average 3. (b) The disagreement $\|x(t) - 3 \cdot \mathbf{1}\|$ on a logarithmic scale, starting from $\sqrt{14} \approx 3.74$: on the topology of Example 2.10 it decays at the rate $\lambda_2 = 1$, and on the bare path P_4 left by deleting the chord 1–3 at the rate $\lambda_2 = 2 - \sqrt{2} \approx 0.586$. The chord speeds up consensus.

Remark 2.11 (Dependent entries in Laplacian families). The entries of a Laplacian are *dependent*: each diagonal entry is the negative sum of its row. When Part II treats matrix entries as free parameters, Laplacian families will need this dependency respected. We flag it now and return to it with the diffusively-coupled networks of Chapter 9.

Sources. The state-space material and the solution and stability results follow [2], with [5] covering the same scope as a second reference. The Lyapunov condition is stated as in [6]. Laplacian and spectral graph facts are [7], and the consensus results (convergence to the average, the rate λ_2, and the rooted-spanning-tree condition on a directed graph) are [8, 9].

Exercises

Exercise 2.1 (Direction convention)**.** State the direction convention of Definition 2.3. Then, for a system with $n = 5$ states and one input, say which edge is present when $[A]_{42} \neq 0$, what a nonzero $[A]_{33}$ draws, and which node becomes a driver node when $[B]_{31} \neq 0$.

Exercise 2.2 (From graph to matrix)**.** A three-node network has state edges $1 \to 2$ of weight 4, $3 \to 2$ of weight -1 and $2 \to 3$ of weight 2, together with a self-loop of weight -5 at node 1. One input drives node 3 and one output reads node 1. Write down A, b and c.

Exercise 2.3 (The motor as a graph)**.** Draw the DC motor of Example 2.2 as a graph in the sense of Definition 2.3: list every edge with its weight, say which nodes carry self-loops, and name the driver node and the node read by the output. How many self-loops does the mass–spring–damper of Example 2.1 have by comparison, and why?

Exercise 2.4 (Rank under multiplication)**.** Let

$$M = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix}, \qquad N = \begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 1 \end{bmatrix}.$$

Compute $\operatorname{rank} M$, $\operatorname{rank} N$ and $\operatorname{rank}(MN)$. Can a product of a 3×2 and a 2×3 matrix ever have rank 3? Justify from Section 2.4.

Exercise 2.5 (Hurwitz and Lyapunov)**.** Let $A = \begin{bmatrix} 0 & 1 \\ -3 & -4 \end{bmatrix}$. (a) Compute its eigenvalues and eigenvectors and decide whether A is Hurwitz. (b) Solve $A^\top M + MA = -I$ and confirm $M \succ 0$ from its leading principal minors.

Exercise 2.6 (Classifying definiteness)**.** Classify each matrix as positive definite, positive semidefinite but not definite, or indefinite:

$$\begin{bmatrix} 3 & 1 \\ 1 & 3 \end{bmatrix}, \qquad \begin{bmatrix} 4 & -2 \\ -2 & 1 \end{bmatrix}, \qquad \begin{bmatrix} 1 & 3 \\ 3 & 1 \end{bmatrix}.$$

For the semidefinite one, give a nonzero direction along which the quadratic form vanishes.

Exercise 2.7 (Sampling is exact)**.** Example 2.7 states that sampling with period T gives $A_d = e^{AT}$ exactly rather than approximately. Explain why, starting from Eq. (2.2).

Exercise 2.8 (Laplacian of four agents)**.** For the four agents of Example 2.10: verify $\mathcal{L}\mathbf{1} = 0$ from the rows, give $\operatorname{rank} \mathcal{L}$ and a basis of $\ker \mathcal{L}$, and read off λ_2. With $x(0) = (1, 3, 2, 6)^\top$, what value does Eq. (2.3) converge to?

Exercise 2.9 (A second chord)**.** The topology of Example 2.10 is the path 1–2–3–4 carrying the chord 1–3. Add the second chord 2–4 as well. Write the Laplacian of all three graphs (the bare path, the path with one chord, the path with two), compute the eigenvalues of each, and rank the three topologies by the rate λ_2 at which the disagreement of Eq. (2.3) decays. Check every spectrum against $\operatorname{tr} \mathcal{L}$, which counts each edge twice.

Exercise 2.10 (Dependent Laplacian entries)**.** Remark 2.11 records that the entries of a Laplacian are dependent. State the dependency, and use it to recover the diagonal of the Laplacian of Example 2.10 from its off-diagonal entries alone.

Chapter 3

Controllability and Observability

The central topic of Part I. Controllability asks whether the input can steer the state anywhere, and observability asks whether the output determines the state. Both reduce to rank conditions. The chapter ends on a deliberate open question: the conclusion of the rank test can flip when a single entry of A changes value, which is a serious problem in a network whose weights are uncertain.

3.1 Controllability

Definition 3.1 (Controllability)**.** The pair (A, B) is *controllable* if for every initial state $x(0) = x_0$ and every target x_f there exist a finite time t_f and an input $u(\cdot)$ with $x(t_f) = x_f$.

Substituting the solution formula Eq. (2.2) shows that everything hinges on which states the zero-state term can reach (a spanning question, as promised in Section 2.4). The fundamental answer is algebraic:

Theorem 3.2 (Kalman rank condition)**.** (A, B) *is controllable iff*

$$\operatorname{rank} \mathcal{C} = \operatorname{rank} \begin{bmatrix} B & AB & \cdots & A^{n-1}B \end{bmatrix} = n. \tag{3.1}$$

Powers beyond A^{n-1} add nothing, by Cayley–Hamilton, which is the reason for the truncation. When the rank falls short of n, the input still steers the state throughout a part of the state space:

Definition 3.3 (Controllable subspace)**.** The set of states reachable from the origin is the column space $\operatorname{Im} \mathcal{C}$, called the *controllable subspace* of (A, B). The pair is controllable iff $\operatorname{Im} \mathcal{C} = \mathbb{R}^n$.

Everything in Part II is a re-reading of Eq. (3.1) and of this subspace, whose dimension and position under parameter uncertainty occupy Chapters 6 and 8.

A second characterization refines the yes/no of Theorem 3.2 into *how hard*:

Theorem 3.4 (Gramian test)**.** (A, B) *is controllable iff the* controllability Gramian

$$W(t) = \int_0^t e^{A\tau} B B^\top e^{A^\top \tau} \, d\tau$$

is nonsingular for some (equivalently every) $t > 0$. In that case the minimum-energy input reaching x_f from the origin has energy $x_f^\top W(t_f)^{-1} x_f$.

The Gramian is symmetric positive semidefinite (Section 2.6). The letter W denotes it here and in Chapter 10, and is unrelated to the weight matrix of Section 2.9. Its energy reading (*how hard* each direction is to reach, not just *whether* it is reachable) returns as the metric for input placement in Chapter 10. For Hurwitz A the infinite-horizon Gramian solves the Lyapunov equation $AW + WA^\top = -BB^\top$, tying it to Theorem 2.8.

Example 3.5 (Energy on a two-node chain)**.** Take $u \to 1 \to 2$ with unit weights, $A = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}$, $b = e_1$, and horizon $T = 1$. Since $A^2 = 0$ the exponential terminates, $e^{A\tau}b = (1, \tau)^\top$, and the integral is exact:

$$W(1) = \int_0^1 \begin{bmatrix} 1 \\ \tau \end{bmatrix} \begin{bmatrix} 1 & \tau \end{bmatrix} d\tau = \begin{bmatrix} 1 & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{3} \end{bmatrix}, \qquad \det W(1) = \tfrac{1}{12} \neq 0.$$

The pair is controllable, as Theorem 3.2 also reports. What the Gramian adds is the cost: by Theorem 3.4 the minimum energy to reach x_f from the origin is $x_f^\top W(1)^{-1} x_f$, with $W(1)^{-1} = \begin{bmatrix} 4 & -6 \\ -6 & 12 \end{bmatrix}$, so reaching the driver node costs 4 while reaching node 2, one edge further from the input, costs 12. Both directions are reachable, but they do not cost the same. That difference is what Chapter 10 uses to choose between input placements.

Example 3.6 (Mass–spring–damper)**.** For Example 2.1, $\mathcal{C} = \begin{bmatrix} 0 & \frac{1}{m} \\ \frac{1}{m} & -\frac{c}{m^2} \end{bmatrix}$, $\det \mathcal{C} = -\frac{1}{m^2} \neq 0$: a force input controls both position and velocity, for every m, c, k.

Example 3.7 (The four-node network)**.** For Example 2.4 the columns of $\mathcal{C}$ are computed by walking the graph of Fig. 2.1:

$$b = e_1, \quad Ab = \begin{bmatrix} 0 \\ 2 \\ 0 \\ 0 \end{bmatrix}, \quad A^2 b = \begin{bmatrix} 0 \\ 0 \\ 2 \\ 6 \end{bmatrix}, \quad A^3 b = \begin{bmatrix} -2 \\ 0 \\ 0 \\ 2 \end{bmatrix}.$$

The input's influence spreads along walks: node 1 at step 0, node 2 at step 1 (via $1 \to 2$, weight 2), nodes 3 and 4 at step 2 (via $1 \to 2 \to 3$ and $1 \to 2 \to 4$, weights $2 \cdot 1$ and $2 \cdot 3$), and so on. Here $\det \mathcal{C} = 8 \neq 0$: controllable. The observation that each column entry is a sum of weight products is no accident, and it becomes the key identity of Chapter 4.

3.2 PBH Tests

Theorem 3.8 (PBH)**.** *(A, B) is controllable iff* $\operatorname{rank}[\, A - \lambda I \;\; B \,] = n$ *for every eigenvalue λ of A. Equivalently, (A, B) is controllable iff no left eigenvector of A is orthogonal to every column of B.*

The modal reading: an uncontrollable system has a mode (a direction z with $z^\top A = \lambda z^\top$) that the input cannot excite ($z^\top B = 0$). PBH localizes the failure that the Kalman test only detects.

Example 3.9 (Symmetry destroys controllability)**.** Drive the center of the twin star of Fig. 3.1:

$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \qquad b = e_1.$$

The vector $z = (0, 1, -1)$ satisfies $z^\top A = 0$ and $z^\top b = 0$: the *antisymmetric* mode of the two end nodes $2, 3$ (Definition 4.5) is uncontrollable from the input, which pushes both of them identically. Indeed $\mathcal{C} = [\, e_1,\ (0,1,1)^\top,\ 2e_1 \,]$ has rank 2. No realization (Definition 4.1) fixes this: the failure is caused by the twin *structure*, a theme Part II makes precise.

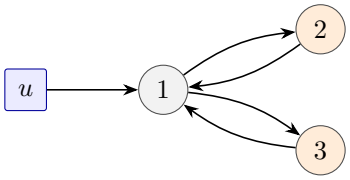


Figure 3.1: Example 3.9: the twin end nodes (shaded) form a symmetric pair, so the input at the center cannot tell them apart and the difference $x_2 - x_3$ is uncontrollable.

3.3 Observability and Duality

Definition 3.10 (Observability)**.** The pair (A, C) is *observable* if the initial state $x(0)$ is determined by the input and output histories on some finite interval.

Theorem 3.11 (Rank test and duality)**.** *(A, C) is observable iff*

$$\operatorname{rank} \mathcal{O} \;=\; \operatorname{rank} \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix} \;=\; n.$$

Equivalently, (A, C) is observable iff $(A^\top, C^\top)$ is controllable.

Duality is exact and complete: every controllability result of this book has an observability dual obtained by transposing (on the graph, by reversing every edge and exchanging the roles of inputs and outputs). We state it once here and use it at the end of the book (Chapter 10).

Example 3.12 (The four-node network, observed)**.** For Example 2.4 with $y = x_4$,

$$\mathcal{O} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 3 & 1 & 0 \\ 6 & 1 & 0 & 0 \\ 2 & 0 & -6 & 0 \end{bmatrix}, \qquad \det \mathcal{O} = -106 \neq 0 :$$

observable. On the graph: every state node has a directed path *to* the output node (4 directly, $2 \to 4$, $3 \to 4$, $1 \to 2 \to 4$), the mirror image of the walk picture of Example 3.7, with all arrows reversed.

A closing remark, stated without development: the Kalman decomposition sorts the state space by controllability and observability, and the transfer matrix depends only on the part that is both, which is exactly why internal stability is stronger than BIBO (Section 2.8).

3.4 Fragility of Numerical Tests

Every test in this chapter used the *numbers* in (A, B). The closing example shows how fragile that reliance is.

Example 3.13 (Same graph, opposite conclusions)**.** Two agents, coupled both ways, each with a self-loop, one input feeding both (Fig. 3.2):

$$A_\circ = \begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}, \quad A_\bullet = \begin{bmatrix} 1 & 2 \\ 3 & 2 \end{bmatrix}, \qquad b = \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

For $b = (1,1)^\top$ one computes $\det[\, b \;\; Ab \,] = a_{21} + a_{22} - a_{11} - a_{12}$, so

$$\begin{aligned} A_\circ &: \ \det \mathcal{C} = 1 + 2 - 1 - 2 = 0 && \text{(uncontrollable)}, \\ A_\bullet &: \ \det \mathcal{C} = 3 + 2 - 1 - 2 = 2 && \text{(controllable)}. \end{aligned}$$

The two systems have the *same* graph (same nodes, same edges) and differ in a single edge weight, 1 versus 3.

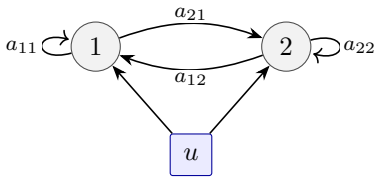


Figure 3.2: Example 3.13: one graph, two realizations (Definition 4.1), opposite controllability conclusions. The graph alone cannot decide, but it can say *when* the numbers matter, which is the subject of Part II.

In a laboratory plant this is a minor issue: the parameters are measured, and non-generic coincidences like $a_{21} + a_{22} = a_{11} + a_{12}$ do not survive a datasheet. In a network the situation is reversed: the weights are exactly what we do not know, so a single rank test computed from nominal values guarantees nothing. What we do know is the pattern, the graph of Definition 2.3 with its numbers erased (Definition 4.3). The question that opens Part II is therefore: *what can the pattern alone guarantee?* Example 3.9 already hints that the answer is not "nothing": some failures are determined by the graph alone.

Sources. [2]. The closing discussion follows [10, 3].

Exercises

Exercise 3.1 (Why the powers stop)**.** State the Kalman rank condition, and explain why the powers A^k with $k \geq n$ contribute no columns to $\mathcal{C}$.

Exercise 3.2 (A weight set to zero)**.** In the four-node network of Example 2.4, set the weight of the edge $3 \to 4$ to zero and leave the other four weights unchanged. Recompute the four columns of $\mathcal{C}$, give $\det \mathcal{C}$ and $\operatorname{rank} \mathcal{C}$, and write the controllable subspace of Definition 3.3 as a span.

Exercise 3.3 (Energy on a longer chain)**.** Extend Example 3.5 to the chain $u \to 1 \to 2 \to 3$ with unit weights, so that $[A]_{21} = [A]_{32} = 1$ and $b = e_1$. Show that $e^{A\tau} b = (1, \tau, \tau^2/2)^\top$, compute $W(1)$ and $\det W(1)$, and give the minimum energy required to reach each of e_1, e_2, e_3 from the origin. Compare with the values 4 and 12 of the two-node chain. [Hint: $A^3 = 0$.]

Exercise 3.4 (PBH on the twin star)**.** For the twin star of Example 3.9, compute the eigenvalues of A and evaluate $\operatorname{rank}[\, A - \lambda I \quad b \,]$ at each of them. At which eigenvalue does Theorem 3.8 fail, and is the outcome consistent with $\operatorname{rank} \mathcal{C} = 2$?

Exercise 3.5 (The uncontrollable weight)**.** In the family of Example 3.13 take $a_{11} = 1$, $a_{12} = 3$, $a_{22} = 2$ and $b = (1, 1)^\top$, leaving a_{21} free. Determine the unique value of a_{21} for which the pair is uncontrollable, and at that value give the eigenvalue λ and left eigenvector z that Theorem 3.8 exhibits.

Exercise 3.6 (Observing position or velocity)**.** For the mass–spring–damper of Example 2.1, compute $\mathcal{O}$ and $\det \mathcal{O}$ (i) with the position measured, $C = \begin{bmatrix} 1 & 0 \end{bmatrix}$, and (ii) with the velocity measured instead, $C = \begin{bmatrix} 0 & 1 \end{bmatrix}$. For which parameter values is the second one observable, and what does that condition mean physically?

Exercise 3.7 (Moving the output)**.** Keep the four-node network of Example 2.4 but move the output to node 3, so that $c = e_3^\top$. Compute $\mathcal{O}$, give $\det \mathcal{O}$ and $\operatorname{rank} \mathcal{O}$, and account for the outcome from the graph of Fig. 2.1.

Exercise 3.8 (Duality on the graph)**.** State the dual of Theorem 3.2 for observability, and describe the operation on the graph of Definition 2.3 that turns a controllability question into the corresponding observability question.

Exercise 3.9 (Laboratory plant against network)**.** Example 3.13 reaches opposite conclusions on one graph. Explain in two or three sentences why this is a minor issue for a laboratory plant but not for a network, and state the question with which Part II opens.

Part II

Controllability of Structured Networks: Analysis

Chapter 4

Structured Networks and Pattern Graphs

Part II opens by changing what we assume: the zero/non-zero pattern of the system matrix is reliable, its numbers are not. This chapter builds the vocabulary for that setting. It introduces the family of a pattern and the pattern graph, and collects in a reference section the graph vocabulary the analysis chapters draw on, which may be read on demand. It then derives the identity showing that the controllability matrix was a graph object all along: powers of the adjacency matrix enumerate walks, and multiplying by B keeps exactly the walks starting at the inputs. The chapter closes with the two kinds of entry, single-term and multi-term, that the identity produces.

4.1 Patterns, Not Numbers

Definition 4.1 (Family)**.** A *pattern matrix* A is a matrix each of whose entries is either structurally zero or a free non-zero parameter. Its *family* $\mathcal{Q}(A)$ is the set of all matrices A_F with $[A_F]_{ij} \neq 0$ exactly where $[A]_{ij} \neq 0$. A member $A_F \in \mathcal{Q}(A)$ is called a *realization* of the pattern. The free non-zero entries are the *network parameters*. A third entry type is used only where a criterion asks for it: an *arbitrary* entry, free to take any value, zero included. A pattern with arbitrary entries generates the larger family in which those positions are unconstrained. The free diagonal of Theorem 7.14 is of that kind. Unless said otherwise every entry in this book is structurally zero or free non-zero.

Example 4.2 (The flip system, as a family)**.** The two matrices of Example 3.13 are two members of one family,

$$A_F = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \qquad a_{ij} \neq 0 \text{ arbitrary},$$

and the question left open there (what survives when only the pattern is known?) becomes: which properties hold *across* the family $\mathcal{Q}(A)$? Likewise the four-node network of Example 2.4 generates the family with free parameters $a_{21}, a_{32}, a_{42}, a_{13}, a_{43}$. The numbers used in Chapter 3 were one member among infinitely many.

At a single realization, everything of Part I applies verbatim: the structured setting strictly contains the classical one. B is held fixed and binary, and the standing convention of Part II is that, unless said otherwise, each input attaches to a *single* driver node. The flip family, whose one input feeds both nodes, is the flagged exception. For single-attachment columns the non-zero magnitudes never affect controllability (rescaling a column of B merely rescales columns of $\mathcal{C}$), so only *where* inputs attach matters. A multi-attachment column, by contrast, mixes weights whose ratios can decide the outcome.

4.2 Pattern Graph

Every realization $A_F \in \mathcal{Q}(A)$ has the graph of Definition 2.3 with its own weights, but all realizations share the same *unweighted* picture.

Definition 4.3 (Pattern graph)**.** The *pattern graph* of a family is the graph of Definition 2.3 with the edge weights erased: one graph, all realizations.

4.3 Graph-Theoretic Background

Part II reasons on the pattern graph, so we collect in one place the graph vocabulary the analysis chapters draw on. The definitions are elementary and stated once here, and the running four-node pattern (Fig. 4.1) illustrates them. We spend an example only where a notion is genuinely easy to misread: in/out orientation, disjointness, and the layered structure of acyclic patterns. The specialized *criteria* built from this vocabulary stay in the chapters that need them.

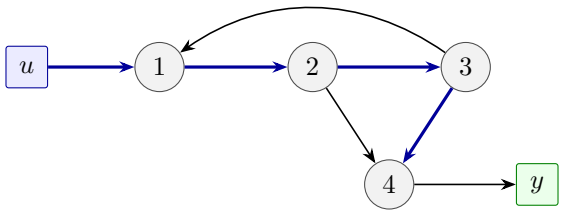


Figure 4.1: The pattern graph of Example 2.4: same picture as Fig. 2.1, numbers erased. It is the running example for this section. The stem (Definition 4.9) $u{\to}1{\to}2{\to}3{\to}4$ (bold) covers all state nodes.

4.3.1 Directed Graphs, Undirected Graphs, and Neighbors

The directed graph of a linear system was fixed once in Definition 2.3: a node for every state, input, and output, and a directed edge (j, i) (drawn $j \to i$) whenever $[A]_{ij} \neq 0$, so that an edge records that j influences i. Two structural specializations recur.

Definition 4.4 (Undirected graph)**.** A pattern is *undirected* (or *symmetric*) if its edge relation is symmetric on the state nodes: $(j, i) \in \mathcal{E}$ iff $(i, j) \in \mathcal{E}$ for all $i, j \in \mathcal{V}^S$. An *undirected edge* between i and j, drawn $i - j$, is then shorthand for the pair of opposite directed edges $i \to j$ and $j \to i$. Algebraically it is the symmetric pair $[A]_{ij}, [A]_{ji} \neq 0$. A *self-loop* at i, an edge $i \to i$, corresponds to a non-zero diagonal entry $[A]_{ii}$.

Undirected patterns model bidirectional coupling: the four-robot network and the Laplacian consensus dynamics of Section 2.9 are the book's standing example, and self-loops (free diagonal entries) return as the hypothesis of the zero-forcing test (Theorem 7.14).

Definition 4.5 (In- and out-neighbors)**.** For a node v, its *out-neighbors* $N^{\text{out}}(v) = \{w : (v, w) \in \mathcal{E}\}$ are the nodes v points to, and its *in-neighbors* $N^{\text{in}}(v) = \{w : (w, v) \in \mathcal{E}\}$ the nodes that point to v. For a set $S \subseteq \mathcal{V}$ the operators act setwise, $N^{\text{out}}(S) = \bigcup_{v \in S} N^{\text{out}}(v)$ and $N^{\text{in}}(S) = \bigcup_{v \in S} N^{\text{in}}(v)$. On an undirected pattern the two coincide on the state nodes, written $N(\cdot)$, and a state node with exactly one neighbor among them, $|N(v) \cap \mathcal{V}^S| = 1$, is an *end node*.

Reading edges the wrong way around is a recurring source of error. Under the transpose convention (Definition 2.3) an out-neighbor of v is a node v *feeds*, carried by v's *column* of A, not its row, so one explicit count is worked out here.

Example 4.6 (In- and out-neighbors on the four-node network)**.** The state edges of Fig. 4.1 are $1 \to 2$, $2 \to 3$, $2 \to 4$, $3 \to 1$, $3 \to 4$. Take the set $S = \{3, 4\}$. Its in-neighbors are the nodes with an edge *into* S: node 2 (via $2 \to 3$ and $2 \to 4$) and node 3 (via $3 \to 4$), so

$$N^{\text{in}}(\{3, 4\}) = \{2, 3\}.$$

Its out-neighbors are the nodes S *feeds*: node 3 reaches 1 and 4, while node 4 reaches only the output y. Among state nodes,

$$N^{\text{out}}(\{3, 4\}) = \{1, 4\}.$$

The two sets differ, $\{2, 3\}$ against $\{1, 4\}$, and node 3 belongs to *both*, receiving from 2 and sending to 4. The whole-graph criteria of Chapter 7 are phrased in exactly this in/out vocabulary ("an edge into S", "an out-neighbor inside S"), so the orientation must be read carefully.

4.3.2 Walks, Paths, Cycles, and Distance

Definition 4.7 (Walk, path, cycle)**.** A *walk* of length k is a sequence of k consecutive directed edges, and its nodes may repeat. The empty sequence is a walk of length zero from each node to itself. A *path* is a walk that visits no node twice, and a *cycle* is a path whose final edge returns to its start node. The degenerate case is a self-loop $i \to i$, a cycle of length one.

The distinction matters: the identity of Section 4.4 counts *walks*, which may loop around a cycle, whereas the covering objects of every controllability criterion are *paths*. The walk $u \to 1 \to 2 \to 3 \to 1$ of Example 4.15 revisits node 1 and so is not a path. Walks and paths differ exactly at cycles.

Definition 4.8 (Distance and shortest path)**.** The *distance* from v to w is the length of a shortest directed walk from v to w (infinite if none exists). A walk attaining it repeats no node and is therefore a *shortest path*. The distance from the input nodes to a state node is its *input-distance*.

Distance underwrites two later bounds: the shortest-stem lower bound on $\dim \mathcal{C}_\Gamma$ (Definition 6.1 and Section 6.4) and the shortest-path criterion for SSC nodes (Theorem 8.17).

4.3.3 Stems, Branches, and Reachability

The input and output nodes single out two dual classes of paths.

Definition 4.9 (Stem)**.** A *stem* is a path originating at an input node.

Definition 4.10 (Branch)**.** A *branch* is a path terminating at an output node. It is the observability dual of a stem, obtained by reversing every edge and exchanging the roles of inputs and outputs.

Stems carry the controllability results of Parts II–III. Branches carry their sensing counterpart and surface when everything is dualized in Section 10.3. On the four-node network the stem $u{\to}1{\to}2{\to}3{\to}4$ runs from the input across every state node, and the path $1{\to}2{\to}3{\to}4{\to}y$ is the matching branch into the single output.

Definition 4.11 (Induced subgraph)**.** The subgraph *induced* by a node set $U \subseteq \mathcal{V}$ keeps the nodes of U and every edge of the pattern with both endpoints in U.

Before specializing to the inputs, the general notion: which nodes can be reached from where.

Definition 4.12 (Reachable set)**.** For any subset $S \subseteq \mathcal{V}$ of nodes, the *reachable set* of S is

$$R(S) = \{\, w \in \mathcal{V} : \text{there exist } v \in S \text{ and a directed path } v \to \cdots \to w \,\} \;\supseteq\; S,$$

the inclusion holding because the empty path leaves each $v \in S$ where it is. A node w is *reachable from* S if $w \in R(S)$. Walks may be used in place of paths without changing $R(S)$, since every walk from v to w contains a path from v to w.

Reachability here is a property of the *graph* (which nodes can be reached along the edges), not of the state space. The reachable *states* of Definition 3.3 are the vector-space notion for which graph reachability is necessary but not sufficient.

Definition 4.13 (Input-connectedness)**.** A state node is *reachable* if some stem reaches it or, equivalently, if it lies in $R(\mathcal{V}^I)$, the reachable set of the input nodes. A pattern in which every state node is reachable, i.e.

$$\mathcal{V}^S \subseteq R(\mathcal{V}^I),$$

is called *input-connected*, and its *reachable part* is the subgraph induced by $R(\mathcal{V}^I)$, the reachable state nodes together with the input nodes.

Remark 4.14 (Terminology). Some references, and earlier drafts of these notes, call an input-connected pattern *accessible* and a reachable node an *accessible* node. The two vocabularies ask the same picture-level question (is there a path from an input to this node?), and *input-connected* is the term used throughout this book.

A state node outside $R(\mathcal{V}^I)$ contributes nothing to controllability: its row of $\mathcal{C}_F$ vanishes identically (Proposition 4.24). Several criteria are therefore stated on the reachable part, and dropping the restriction breaks them, as Example 7.4 shows in full.

Example 4.15 (Stems of the four-node pattern)**.** The pattern graph of Example 2.4 (Fig. 4.1) has the stems

$$u{\to}1, \quad u{\to}1{\to}2, \quad u{\to}1{\to}2{\to}3, \quad u{\to}1{\to}2{\to}4, \quad u{\to}1{\to}2{\to}3{\to}4,$$

the last one covering every state node, a fact that Chapter 6 will turn into a sufficient condition for *structural* controllability. All four state nodes are reachable, $R(\{u\}) = \{u, 1, 2, 3, 4\}$, so the pattern is input-connected. Note the walk $u \to 1 \to 2 \to 3 \to 1$ is not a stem (node 1 repeats).

4.3.4 Disjoint Paths and Stems

Definition 4.16 (Disjointness)**.** Write $V(p)$ and $E(p)$ for the node set and the edge set of a path p. Two paths, in particular two stems, are *node-disjoint*, or simply *disjoint*, if they share no node, $V(p) \cap V(q) = \emptyset$. A set of paths is disjoint if its members are pairwise disjoint.

Remark 4.17 (Node- versus edge-disjoint). A weaker notion exists: two paths are *edge-disjoint* if they share no edge, $E(p) \cap E(q) = \emptyset$, nodes allowed. Node-disjoint implies edge-disjoint, never the converse: the stems $u \to 1 \to 2$ and $u \to 3 \to 4$ use entirely different edges yet share the input u. They are edge-disjoint, not node-disjoint. Throughout this book, and in most of the structured-systems literature, an unqualified *disjoint* always means *node*-disjoint. It is the node version that carries the control meaning developed below: stems that share no state node.

Disjointness is where single-input patterns meet a hard limit. Two stems originating at the *same* input already share that input node, so they are never disjoint. A set of m pairwise disjoint stems therefore requires m distinct input nodes.

Example 4.18 (Two stems that are not disjoint)**.** Consider a single input driving node 1, from which two paths run along $1 \to 2 \to 4$ and $1 \to 3 \to 4$ and rejoin at node 4, the diamond of Fig. 4.2. Read as stems, the two paths $u \to 1 \to 2 \to 4$ and $u \to 1 \to 3 \to 4$ share the input, node 1, *and* node 4: disjoint in neither. With one input a disjoint set can therefore hold at most one stem, and any further nodes must be picked up by disjoint *cycles*. This is exactly why the generic-dimension count of Theorem 6.4 is stated for disjoint stems *and cycles*, and why the diamond's lone input caps its cover at three of four nodes (Example 6.5).

4.3.5 Connectivity

Definition 4.19 (Connectivity notions)**.** A directed pattern is *strongly connected* if it has a directed walk from every node to every other node. Its *underlying undirected graph* (edges with direction forgotten) is *connected* if it has an undirected path between every pair of nodes, and is a *tree* if it is connected and acyclic. A *rooted spanning tree* of a directed pattern is a subgraph that touches every node and in which one node, the *root*, reaches all others by directed paths (equivalently, an acyclic spanning subgraph in which the root has no in-edge and every other node exactly one).

These notions govern the network dynamics of Section 2.9: an undirected consensus network reaches average agreement iff it is connected, and a directed one reaches weighted agreement iff it has a rooted spanning tree.

4.3.6 Acyclic and Hierarchical Structure

Patterns without cycles support the sharpest results in the book: on them every walk is a path, and no length can be inflated by looping.

Definition 4.20 (Directed acyclic graph)**.** A pattern is a *directed acyclic graph* (DAG) if it has no directed cycle of any length $k \geq 1$. In particular a self-loop (Definition 4.4) is a cycle of

length one, so a DAG carries no self-loop and its diagonal is structurally zero: $[A]_{ii} = 0$ for every i. A *source* is a state node with no in-edges from other state nodes, and a *sink* is a state node with no out-edges to other state nodes. The input and output nodes are left out of the count deliberately, so that a driver node counts as a source and a state node feeding only an output counts as a sink. (The papers this book draws on call a sink a *terminal node*. One of them reserves that name instead for a state node with a single neighbor, an end node in the sense of Definition 4.5, so *sink* is the name used here.)

On a DAG the state matrix is nilpotent (no walk exceeds $n-1$ edges, so $A_F^n = 0$, which Remark 10.1 takes up), and the nodes fall into layers by input-distance. When those layers are clean the pattern is given a stronger name.

Definition 4.21 (Distance layers and hierarchical DAG)**.** The *distance layers* of a pattern are the sets ℓ_k of state nodes at input-distance exactly k (Definition 4.8). The symbol ℓ_k is used for them in Chapter 8 and the appendices as well. A DAG is a *hierarchical DAG* (HDAG) if the sets of nodes reachable from the inputs in exactly k steps are disjoint across k, so that every walk into a given node has one common length and the reachable state nodes are partitioned into their distance layers $\ell_1, \ell_2, \dots$ (on an input-connected pattern, Definition 4.13, that is all of $\mathcal{V}^S$).

Example 4.22 (The diamond is hierarchical, the four-node network is not)**.** The diamond of Fig. 4.2 (input at node 1, then $1 \to 2$, $1 \to 3$, $2 \to 4$, $3 \to 4$) lays its four nodes into clean distance layers

$$\ell_1 = \{1\}, \qquad \ell_2 = \{2, 3\}, \qquad \ell_3 = \{4\},$$

because *every* walk from the input to node 4 has length three: it is an HDAG. The four-node network of Fig. 4.1 is not, even setting its cycle aside: node 4 is reached from the input both in three steps ($u \to 1 \to 2 \to 4$) and in four ($u \to 1 \to 2 \to 3 \to 4$), so no single distance layer can hold it. This clean layering is what makes the worst-case dimension $\dim \mathcal{C}_\Gamma$ exactly computable on HDAGs (Section 6.4), while no closed formula is available in general.

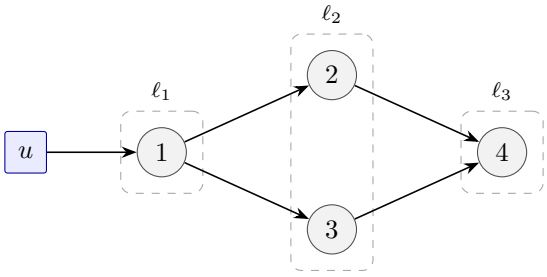


Figure 4.2: The diamond of Example 4.22 arranged into distance layers $\ell_1 = \{1\}$, $\ell_2 = \{2, 3\}$, $\ell_3 = \{4\}$: every walk from the input to a node has a single length, so the pattern is a hierarchical DAG. The four-node network of Fig. 4.1, whose node 4 sits at two different distances, is not.

The vocabulary above is the shared background. Further terms are introduced with the criterion that needs them: *dilation* and *matchings* for structural controllability (Definitions 7.1 and 7.5), *dedicated* and *sharing* nodes and *zero forcing* for strong structural controllability (Definitions 7.7 and 7.13), and the *matched nodes* of a hierarchical DAG and the *control paths* that localize those tests to individual nodes (Theorems 8.10 and 8.17). Table 4.1 collects the recurring terms with pointers.

Table 4.1: Graph vocabulary of Part II: the shared background of this section (top) and the criterion-specific terms introduced later (bottom).

Term	Meaning (in brief)	Defined in
directed graph	edge $j \to i$ when $[A]_{ij} \neq 0$	Definition 2.3
undirected graph	symmetric pair of edges	Definition 4.4
in/out-neighbor	nodes into / out of node	Definition 4.5
end node	exactly one neighbor	Definition 4.5
walk	edge sequence, nodes repeatable	Definition 4.7
path	walk without repeated node	Definition 4.7
cycle	path returning to start	Definition 4.7
distance	length of shortest walk	Definition 4.8
stem	path starting at input	Definition 4.9
branch	path ending at output	Definition 4.10
induced subgraph	nodes kept with internal edges	Definition 4.11
reachable set $R(S)$	nodes reached by paths from S	Definition 4.12
input-connected	$\mathcal{V}^S \subseteq R(\mathcal{V}^I)$	Definition 4.13
disjoint paths	sharing no node	Definition 4.16
strongly connected	every node reaches every	Definition 4.19
rooted spanning tree	one root reaches all	Definition 4.19
DAG	no directed cycle	Definition 4.20
source / sink	no state in-edges / no state out-edges	Definition 4.20
hierarchical DAG	clean distance layers	Definition 4.21
shortest stem	unique walk to each node at its input-distance	Theorem B.1
dilation	set with too few in-neighbors	Definition 7.1
tail / head	of edge $j \to i$: j / i	Definition 7.5
matching	no two edges share a tail or a head	Definition 7.5
dedicated node	exactly one edge into set	Definition 7.7
sharing node	two or more edges into set	Definition 7.7
zero forcing set	drivers force whole graph	Definition 7.13
matched node (layer)	on an HDAG, in every maximum disjoint-stem set	Theorem 8.10
SSC node	every subset containing it is controllable	Definition 8.16
control path	shortest-path graph, input to sink	Theorem 8.17
polytree	tree once directions are ignored	Theorem 8.17

4.4 Graph-Theoretic Meaning of the Controllability Matrix

The identity underlying all of Part II is assembled in two steps, first a fact of pure graph theory and then one matrix multiplication, after which its entries fall into two kinds.

4.4.1 Powers of Adjacency Matrix Count Walks

Lemma 4.23 (Walk enumeration)**.** *Interpreting $A_F \in \mathcal{Q}(A)$ as a weighted adjacency matrix, $[A_F^k]_{lw}$ equals the sum, over all k-step walks from node w to node l on the pattern graph, of the products of the edge weights along the walk. For a $0/1$ matrix it counts the walks.*

Proof. Induction on k. For $k = 1$ the statement is the definition of the graph. For the step, $[A_F^{k+1}]_{lw} = \sum_j [A_F]_{lj}[A_F^k]_{jw}$: every $(k{+}1)$-step walk from w to l is a k-step walk from w to some j followed by an edge $j \to l$, and the weight products multiply accordingly. □

4.4.2 Multiplying by B: Walks from Inputs

Proposition 4.24 (Input-walk enumeration)**.** *By Lemma 4.23, and since the columns of B select the driver nodes, the entry $(l, k{+}1)$ of each per-input block of $\mathcal{C}_F = [B, A_F B, \ldots, A_F^{n-1} B]$*

equals the sum of the weight products of all k-step walks from the driver node selected by that input column to node l (the leading input edge carries weight 1, so the count starts at the driver node): the controllability matrix lists, length by length, every walk by which the inputs reach the states.

The enumerated walks may revisit nodes (Example 4.25 below contains a walk that goes once around a cycle), and the repeat-free ones among them are exactly the stems of Definition 4.9 with their input edge dropped.

Example 4.25 (The four-node family, symbolically)**.** For the four-node family generated by Example 2.4 (introduced at the end of Example 4.2),

$$b = e_1, \quad A_F b = a_{21} e_2, \quad A_F^2 b = a_{32}a_{21}\, e_3 + a_{42}a_{21}\, e_4, \quad A_F^3 b = a_{13}a_{32}a_{21}\, e_1 + a_{43}a_{32}a_{21}\, e_4.$$

Each term is a walk read off Fig. 4.1, counted from the driver node 1 (the weight-1 attachment $u{\to}1$ is suppressed). The term a_{21} is the one-step walk $1{\to}2$, and the two terms of $A_F^2 b$ are $1{\to}2{\to}3$ and $1{\to}2{\to}4$. The e_1-term of $A_F^3 b$ is the walk once around the cycle $1{\to}2{\to}3{\to}1$, and its e_4-term the stem $1{\to}2{\to}3{\to}4$. The numerical columns of Example 3.7 are these expressions evaluated at one realization. Every entry here is a *single* product of weights, the case named in Section 4.4.3 below, which Chapter 5 makes decisive. One computation we record for later: expanding $\det \mathcal{C}_F$ along these columns, the only non-vanishing choice picks e_1 from the first, $a_{21}e_2$ from the second, the e_3-term from the third, and the e_4-term from the fourth, so

$$\det \mathcal{C}_F \;=\; a_{21}^3\, a_{32}^2\, a_{43}.$$

This is a single monomial in the network parameters.

4.4.3 Single-Terms and Multi-Terms

By Proposition 4.24 every entry of $\mathcal{C}_F$ is a polynomial in the network parameters, one term per enumerated walk. Two names, both from [11, 12], separate the two levels of that polynomial: the *weight product* (WP) of a walk is the product of the weights of its edges, and the entry $[\mathcal{C}_F]_{i,k+1}$ is the *sum of weight products* (SWP) of all k-step walks from the driver node to node i, steps counted from the driver as in Proposition 4.24. What distinguishes one entry from another is how many walks it sums.

Definition 4.26 (Single-term and multi-term)**.** A non-zero entry of $\mathcal{C}_F$ is a *single-term* if it consists of a single weight product (one monomial, possibly a single edge weight, or the empty product 1 that the driver node carries in the first column), and a *multi-term* if it is a sum of two or more distinct weight products. A weight product carried by more than one walk counts as one term, whatever integer coefficient it acquires. An entry that enumerates no walk is zero, and is neither.

On an acyclic pattern the classification is readable on the graph, with no polynomial to compute. The statement is adapted from [12].

Proposition 4.27 (Composition of an entry)**.** *Let the pattern graph be a DAG (Definition 4.20) with a single driver node. Then the entry $[\mathcal{C}_F]_{i,k+1}$ is zero if no stem reaches node i in k steps from the driver, a single-term if exactly one stem does, and a multi-term if two or more do.*

Proof. On a DAG every walk is a path (a repeated node would close a cycle), so the walks enumerated by Proposition 4.24 are exactly the stems of Definition 4.9 reaching node i, with their input edge dropped. Two distinct paths with the same endpoints use distinct edge sets and so contribute distinct monomials, each entering with coefficient $+1$. None can merge or cancel, and the number of terms is the number of stems. □

Remark 4.28 (Walks against terms). With cycles present the correspondence is walk-based and looser. The $(1,4)$ entry $a_{13}a_{21}a_{32}$ of Example 4.25 is a single-term carried by one walk, once around the cycle $1\to2\to3\to1$, though no stem returns to node 1 at all. Two distinct walks of equal length may also carry the same WP, combining into one monomial with an integer coefficient. What holds in general is that a single-term never vanishes: an entry becomes zero only when distinct monomials cancel at a particular realization, and that takes a multi-term.

The patterns already in hand carry the count from one term to two, and one further pattern takes it to three. For the chain $u\to1\to2\to3$ of Example 5.7,

$$\mathcal{C}_F = \begin{bmatrix} 1 & 0 & 0 \\ 0 & a_{21} & 0 \\ 0 & 0 & a_{21}a_{32} \end{bmatrix},$$

every non-zero entry is a single-term: each node is reached by exactly one stem, at one length. Assembling the columns of Example 4.25 for the four-node pattern of Fig. 4.1,

$$\mathcal{C}_F = \begin{bmatrix} 1 & 0 & 0 & a_{13}a_{21}a_{32} \\ 0 & a_{21} & 0 & 0 \\ 0 & 0 & a_{21}a_{32} & 0 \\ 0 & 0 & a_{21}a_{42} & a_{21}a_{32}a_{43} \end{bmatrix},$$

and every entry is a single-term again, cycle notwithstanding: node 4 is reached in two steps along $1\to2\to4$ and in three along $1\to2\to3\to4$, but stems of different lengths land in different columns and never share an entry. The diamond of Fig. 4.2 is the first pattern here in which two stems do share one entry:

$$\mathcal{C}_F = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & a_{21} & 0 & 0 \\ 0 & a_{31} & 0 & 0 \\ 0 & 0 & a_{21}a_{42} + a_{31}a_{43} & 0 \end{bmatrix}.$$

Its $(4,3)$ entry is a multi-term of two WPs, one from $u\to1\to2\to4$ and one from $u\to1\to3\to4$, the two stems reaching node 4 in the same number of steps. It is the entry whose cancellation leaves the diamond with the worst-case dimension $\dim \mathcal{C}_\Gamma = 2$ computed in Example 6.2. The two step counts of this chapter differ by one and meet in that entry: node 4 lies at input-distance 3, so $\ell_3 = \{4\}$ (Example 4.22), while the entry sits in column 3 because the driver reaches node 4 in two steps. A node at input-distance k carries its entry in column k of $\mathcal{C}_F$, the leading input edge accounting for the difference.

Example 4.29 (Three stems into one node)**.** The pattern of Fig. 4.3 (input at node 1, then

$1\to2$, $1\to3$, $1\to4$, and $2\to5$, $3\to5$, $4\to5$) sends three stems of equal length into node 5:

$$\mathcal{C}_F = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & a_{21} & 0 & 0 & 0 \\ 0 & a_{31} & 0 & 0 & 0 \\ 0 & a_{41} & 0 & 0 & 0 \\ 0 & 0 & a_{21}a_{52} + a_{31}a_{53} + a_{41}a_{54} & 0 & 0 \end{bmatrix}.$$

The $(5,3)$ entry is a multi-term of three WPs, one for each of $u\to1\to2\to5$, $u\to1\to3\to5$ and $u\to1\to4\to5$. Every other non-zero entry is a single-term. Generically the rank is three, and that entry is the only place where it can drop: at $a_{21}a_{52} = a_{31}a_{53} = 1$ and $a_{41}a_{54} = -2$ the sum is zero, the third column vanishes, and the rank is two.

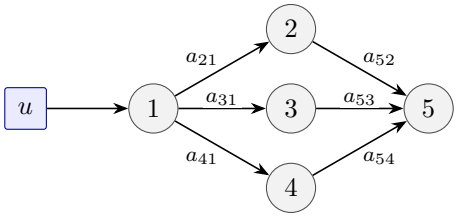


Figure 4.3: The fan of Example 4.29: three stems of equal length into node 5 make the $(5,3)$ entry of $\mathcal{C}_F$ a multi-term of three WPs, while every other non-zero entry is a single-term.

Where the multi-terms sit is itself a property of the picture. Two node classes name it, and the proposition after them locates every multi-term on a hierarchical DAG. Both are adapted from [12].

Definition 4.30 (Integrator and intermediator)**.** A state node with two or more in-neighbors among the state nodes is an *integrator* (a graph classification, unrelated to the dynamic element $1/s$). A state node with exactly one such in-neighbor is an *intermediator* if every stem leading to it passes through at least one integrator.

Proposition 4.31 (Location of the multi-terms)**.** *Let the pattern graph be an HDAG (Definition 4.21) that is input-connected (Definition 4.13) and has a single driver node. The driver is then its only source among the state nodes. Row i of $\mathcal{C}_F$ contains a multi-term if and only if node i is an integrator or an intermediator.*

Proof. On an HDAG every stem reaches i in the same number of steps, so row i carries at most one non-zero entry and Proposition 4.27 reads it as a count of stems. Two or more in-neighbors give two or more stems. With exactly one in-neighbor j, every stem to i extends a stem to j, so the two counts agree. Iterating back to the driver, whose count is one, we find that the count exceeds one exactly when an integrator lies on the way. □

The diamond and the fan are both hierarchical DAGs driven at their only source, so the proposition applies to both: node 4 of the diamond and node 5 of the fan have two and three in-neighbors, are integrators, and carry the multi-terms computed above. An intermediator, by contrast, is fed by a single edge and lies downstream of an integrator, whose multi-term it inherits. Append one edge $4\to5$ of weight a_{54} to the diamond and node 5 becomes one (Fig. 4.4): its only non-zero entry is

$$[\mathcal{C}_F]_{5,4} = a_{54}(a_{21}a_{42} + a_{31}a_{43}),$$

the multi-term of node 4 carried one step further. A node need not collect several edges of its own for its entry to become zero.

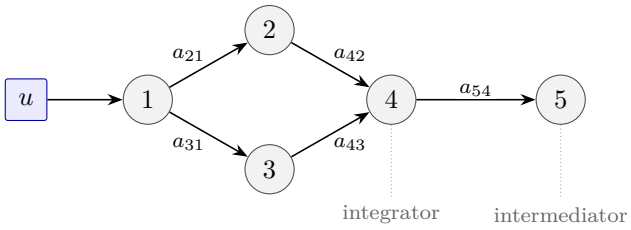


Figure 4.4: One edge 4→5 of weight a_{54} appended to the diamond of Fig. 4.2. Node 4 collects two in-edges and is an integrator. Node 5 has a single in-neighbor, but every stem to it runs through node 4, so it is an intermediator (Definition 4.30) and inherits the multi-term, $[\mathcal{C}_F]_{5,4} = a_{54}(a_{21}a_{42} + a_{31}a_{43})$.

The two classes divide the work of the chapters that follow. A single-term is a product of non-zero network parameters and cannot vanish. Only a multi-term can. Chapter 5 reads $\det \mathcal{C}_F$ through this division, and Chapter 6 measures how much rank a vanishing multi-term can cost. Which multi-terms can never vanish is settled in [12] and stays outside this book. The exact worst-case dimension it yields on hierarchical DAGs is the result Section 6.4 quotes.

4.5 Remark on Outputs and Targets

If only some states matter, a selection matrix C picks them out, and the natural condition becomes $\operatorname{rank}(C\,\mathcal{C}) = q$, the number of selected states. That condition is *target controllability*, the output-level counterpart of everything that follows. We note it here and leave its theory to the literature [13, 14, 15].

4.6 Questions of Part II

With the vocabulary in place, the questions of Part II: when does the rank of $\mathcal{C}_F$ survive the unknown weights (Chapter 5), how much survives when full rank fails (Chapter 6), the same questions asked on the graph (Chapter 7), which individual nodes survive (Chapter 8), and finally, where inputs should be placed (Chapters 9 and 10).

Sources. The pattern and family viewpoint, and the graph vocabulary collected in Section 4.3, follow [16] and the author's [1, 4]. Walk enumeration (Lemma 4.23) is classical algebraic graph theory [7]. Section 4.4.3 (weight products and their sums, the single-term/multi-term distinction, the integrator and intermediator classes, and the location of the multi-terms) is adapted from the author's [12, 11].

Exercises

Exercise 4.1 (Family and realization)**.** Decide true or false, with a one-sentence reason from Definition 4.1 in each case. (a) Two realizations of one family have the same pattern graph. (b) A realization may carry a zero where the pattern has a free non-zero parameter. (c) Any matrix that is zero wherever the pattern is zero is a realization of that pattern.

Exercise 4.2 (In- and out-neighbors)**.** On the four-node pattern of Fig. 4.1, compute $N^{\mathrm{in}}(S)$ and $N^{\mathrm{out}}(S)$ for $S = \{1, 2\}$, and $N^{\mathrm{in}}(\{4\})$, counting state nodes only. Then name the non-zero entries in the second column of A and say which of the two neighbor sets of node 2 that column lists.

Exercise 4.3 (Walks against paths)**.** On the same pattern, list every walk of length three starting at node 1 and say which of them are paths. Then identify the only walk of length four starting at node 1, and conclude that no walk of that length reaches node 4. [Check both counts against A_F^3 and A_F^4 using Lemma 4.23.]

Exercise 4.4 (Distance layers)**.** For each pattern below give the distance layers ℓ_k, then decide from the sets of nodes reachable in exactly k steps which of the three are hierarchical DAGs (Definition 4.21). (a) The fan of Example 4.29. (b) The three-node pattern with driver node 1 and edges $1\to2$, $1\to3$, $2\to3$. (c) The diamond of Example 4.22 with one further edge $4\to5$ appended.

Exercise 4.5 (Single-terms and multi-terms)**.** Append one edge $5\to6$ of weight a_{65} to the fan of Example 4.29. Classify every non-zero entry of the resulting $\mathcal{C}_F$ as a single-term or a multi-term (Definition 4.26) and count the entries that are zero. Confirm the classification of the $(6, 4)$ entry by counting, as in Proposition 4.27, the stems reaching node 6 in three steps from the driver node.

Exercise 4.6 (Integrators and intermediators)**.** Take the fan of Example 4.29 with the edge $5\to6$ of weight a_{65} appended, as in the previous exercise. Classify each of the six state nodes as an integrator, an intermediator, or neither (Definition 4.30), check that the three hypotheses of Proposition 4.31 hold, and name the rows of $\mathcal{C}_F$ that carry a multi-term. [Nodes 2, 3 and 4 each have one in-neighbor. Ask what lies upstream of it.]

Exercise 4.7 (Reachable part)**.** Delete the edge $2\to3$ from the four-node pattern of Fig. 4.1, keeping the driver node 1. Compute $R(\{u\})$, decide whether the pattern is input-connected (Definition 4.13), draw the reachable part, and name the state node whose row of $\mathcal{C}_F$ vanishes identically. Which state node is a source, and which a sink (Definition 4.20)?

Exercise 4.8 (Walks, not stems)**.** Explain from Proposition 4.24 why the $(1, 4)$ entry $a_{13}a_{21}a_{32}$ of the four-node network's $\mathcal{C}_F$ is non-zero although no stem returns to node 1, and say which hypothesis of Proposition 4.27 that pattern fails.

Exercise 4.9 (Which entries can vanish)**.** Decide true or false, with a reason from Definition 4.26 and Remark 4.28. (a) An entry of $\mathcal{C}_F$ that is non-zero at one realization and zero at another is a multi-term. (b) A single-term is non-zero at every realization. (c) An entry that enumerates no walk is a single-term.

Chapter 5

Structural and Strong Structural Controllability: Determinant View

The central definitions of the book, and one object that separates them: the determinant of the controllability matrix, read as a polynomial in the network parameters. It vanishes identically (never controllable), vanishes only on a measure-zero set (structurally controllable), or never vanishes (strongly structurally controllable). Each of the three cases fits on a graph with two or three nodes.

5.1 Almost All versus All

Definition 5.1 (Structural controllability)**.** The family $\mathcal{Q}(A)$ with input matrix B is *structurally controllable* (SC) if the pair (A_F, B) is controllable for almost all realizations $A_F \in \mathcal{Q}(A)$.

Definition 5.2 (Strong structural controllability)**.** The family $\mathcal{Q}(A)$ with input matrix B is *strongly structurally controllable* (SSC) if the pair (A_F, B) is controllable for *all* realizations $A_F \in \mathcal{Q}(A)$.

A single realization recovers the classical notion, and SSC $\Rightarrow$ SC. The open question of Section 3.4 can now be answered in one line.

Example 5.3 (The flip family, resolved)**.** For the family of Example 4.2 with $b = (1, 1)^\top$,

$$\det \mathcal{C}_F = a_{21} + a_{22} - a_{11} - a_{12},$$

a polynomial that is not identically zero but does vanish on the hyperplane $a_{21} + a_{22} = a_{11} + a_{12}$ in parameter space. The family is SC (almost every member controllable) but not SSC (the member $A_\circ$ of Example 3.13 sits exactly on the hyperplane). What looked like a numerical special case in Chapter 3 is a *structural* fact about where, in parameter space, the pattern can fail.

5.2 Determinant as Parameter Polynomial

Proposition 5.4 (Determinant trichotomy)**.** *Let $m = 1$, so that $\mathcal{C}_F$ is square and $\det \mathcal{C}_F$ is a polynomial in the network parameters. Exactly one of the following holds:*

(i) $\det \mathcal{C}_F \equiv 0$*: no realization is controllable.*

(ii) $\det \mathcal{C}_F \not\equiv 0$ *but vanishes at some realization: the family is SC, and the uncontrollable realizations form a non-empty measure-zero set.*

(iii) $\det \mathcal{C}_F \neq 0$ *for every realization: the family is SSC.*

Case (ii) contains the genericity principle (a non-trivial polynomial vanishes only on a measure-zero set, so *one* controllable realization establishes SC) and carries all the content of the proof, given in Section A.1. The three cases are exhibited by three minimal patterns (Fig. 5.1):

Example 5.5 (Never: the twin star)**.** For the star of Example 3.9 as a family (free weights $a_{12}, a_{13}, a_{21}, a_{31}$, input at the center),

$$\mathcal{C}_F = \left[\, e_1, \;\; a_{21}e_2 + a_{31}e_3, \;\; (a_{12}a_{21} + a_{13}a_{31})\, e_1 \,\right] :$$

the first and third columns are parallel for *every* realization, so $\det \mathcal{C}_F \equiv 0$. The symmetry failure of Example 3.9 was never about the numbers: no realization can separate two end nodes fed through a single driver node.

Example 5.6 (Almost all: the flip family)**.** Example 5.3: $\det \mathcal{C}_F \not\equiv 0$ with a non-empty zero set. The family is SC, not SSC.

Example 5.7 (Always: the chain)**.** For the chain $u \to 1 \to 2 \to 3$,

$$\mathcal{C}_F = \left[\, e_1, \;\; a_{21}e_2, \;\; a_{32}a_{21}e_3 \,\right], \qquad \det \mathcal{C}_F = a_{21}^2\, a_{32},$$

a single product of non-zero parameters: it cannot vanish. The chain is SSC (controllable at every realization).

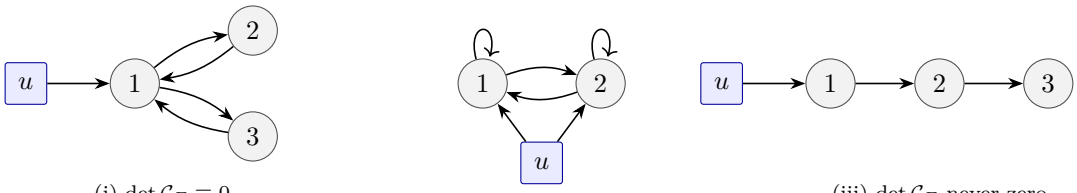


Figure 5.1: The determinant trichotomy on three minimal patterns: Examples 5.5 to 5.7. Each of the three outcomes is a property of the picture, established by one polynomial.

5.3 Reading Determinants on Graphs

Why did the three examples behave so differently? By Proposition 4.24 every entry of $\mathcal{C}_F$ is the SWP of the walks it enumerates, so every non-zero entry is a single-term or a multi-term (Definition 4.26): the former can never vanish, so an entry that becomes zero for some realization is necessarily a multi-term.

The determinant inherits this structure: its expansion terms are signed products of walk weights, one walk per node, and the surviving monomials correspond to *disjoint sets of stems*

and cycles on the pattern graph. Consider the three examples again. The chain's determinant is a single monomial: there is exactly one way to assign walks, the full stem, and nothing can cancel it (SSC). The flip family is the exception flagged in Section 4.1: its one input feeds both nodes, so the disjoint-stem reading does not apply. Its determinant instead expands into the four monomials $a_{21} + a_{22} - a_{11} - a_{12}$ of Example 5.3, each the weight of a single one-step walk, which cancel on $a_{21} + a_{22} = a_{11} + a_{12}$ (SC only). The star admits no such assignment of walks at all (one stem cannot cover two twin end nodes), so every expansion term vanishes identically (never controllable). Making this reading precise, and quantitative, is the subject of the next two chapters: the count of nodes that can be covered measures rank itself (Chapter 6), and the whole-graph criteria of Chapter 7 are this observation turned into necessary and sufficient conditions.

5.4 Multiple Inputs

For $m > 1$ the controllability matrix is $n \times nm$ and the determinant becomes a family of maximum minors. The trichotomy survives with one subtlety worth a remark: strong structural controllability requires that *for every* realization *some* minor is non-zero. The surviving minor may change with the parameters. Exhibiting one minor that never vanishes (a single monomial, as in the single-input examples above) is therefore sufficient, and asks more than the definition does. That is the approach taken wherever this book establishes SSC with several inputs: the diamond's two minimum driver sets each leave one maximum minor equal to a single product of network parameters (Example 9.10). The single-input case remains the teaching case.

5.5 Grid: Algebra Row

The book's running summary (SC/SSC against algebra/graph) begins with its algebra row:

	SC (almost all)	**SSC** (all)
algebra	$\det \mathcal{C}_F \not\equiv 0$	$\det \mathcal{C}_F$ never zero
graph	Chapters 6 and 7	Chapters 6 and 7

Both cells of this row answer yes or no. When the answer is no, that is, when the determinant vanishes identically or vanishes at some realization, the rank does not simply disappear: it settles between a generic maximum and a worst-case minimum. How much survives, and what those two numbers mean, is the subject of Chapter 6.

Sources. SC and SSC in the family form of Definitions 5.1 and 5.2, and the trichotomy of Proposition 5.4, follow the author's [1] and the survey [4]. The two notions originate with Lin [16] and with Mayeda and Yamada [17]. The genericity principle behind case (ii) is standard [10, 18].

Exercises

Exercise 5.1 (Almost all against all)**.** State Definitions 5.1 and 5.2 precisely, explain in one sentence why SSC implies SC, and name the family of this chapter for which the converse fails.

Exercise 5.2 (Where the flip family fails)**.** For the flip family of Example 4.2 with $b = (1, 1)^\top$, compute $\det \mathcal{C}_F$ from $\mathcal{C}_F = [\, b,\ A_F b \,]$ and describe the set of realizations at which the pair is uncontrollable. Decide which of $a_{11} = a_{12} = a_{21} = a_{22} = 1$ and $a_{11} = a_{12} = a_{22} = 1,\ a_{21} = 2$ lies in that set.

Exercise 5.3 (Three patterns, three cases)**.** Compute $\det \mathcal{C}_F$ for each three-node pattern below (driver node 1, $b = e_1$) and place it in case (i), (ii) or (iii) of Proposition 5.4. Pattern (a) has edges 1→2 and 1→3. Pattern (b) has edges 1→2, 1→3, 2→3 and 3→2. Pattern (c) has edges 1→2, 2→3 and 1→3. For the one in case (ii), exhibit a realization at which the pair is uncontrollable.

Exercise 5.4 (Build a pattern of case (iii))**.** Construct a three-node pattern with driver node 1 whose graph contains a directed cycle and whose family is SSC, and establish the claim by computing $\det \mathcal{C}_F$. [Recall from Definition 4.20 that a self-loop counts as a cycle.]

Exercise 5.5 (One monomial)**.** Explain why a determinant equal to a single product of network parameters places a family in case (iii) of Proposition 5.4, and why $\det \mathcal{C}_F \not\equiv 0$ on its own does not. Relate both to the single-term/multi-term distinction of Definition 4.26.

Exercise 5.6 (A multi-term that never vanishes)**.** Take the two-node pattern with self-loops at nodes 1 and 2 and the edge 1→2, driven at node 1. List the three walks of length three from node 1 to node 2, write the sum of their weight products (Lemma 4.23), and show that it is non-zero at every realization. [Complete the square in a_{11}.] Append the edges 2→3 and 3→4, so that this sum becomes the $(2, 4)$ entry of $\mathcal{C}_F$, and state what this says about the converse of the rule, recorded in Section 5.3, that an entry which becomes zero at some realization is necessarily a multi-term.

Exercise 5.7 (A second driver on the twin star)**.** The twin star of Example 5.5 has $\det \mathcal{C}_F \equiv 0$. Attach a second input to node 2, so that $B = [\, e_1,\ e_2 \,]$, and establish that the family is now SSC by exhibiting one maximum minor of $\mathcal{C}_F$ that is a single product of network parameters.

Chapter 6

Two Controllable Subspaces

When full controllability fails, the right question is not yes or no but how much. As the parameters vary, the rank of the controllability matrix moves between a generic maximum and a worst-case minimum, defining two subspaces. Full dimension of each is exactly equivalent to SC and SSC, respectively. Two further subspaces, obtained by intersecting over realizations, are named at the end of the chapter and taken up in Chapter 8. One pattern, the diamond, exhibits all of this: a generic dimension of three, a worst case of two, and a twin pair of nodes never independently controllable.

6.1 Rank Between Two Bounds

Definition 6.1 (SCS and SSCS)**.** $\dim \mathcal{C}_\Lambda$ is the maximum (generic) rank of $\mathcal{C}_F$ over the family, and the *structurally controllable subspace* $\mathcal{C}_\Lambda$ denotes the controllable subspace (Definition 3.3) of any realization attaining it. $\dim \mathcal{C}_\Gamma$ is the minimum (worst-case) rank, and the *strongly structurally controllable subspace* $\mathcal{C}_\Gamma$ the controllable subspace of a realization attaining that. Only the two *dimensions* are invariants of the pattern: the subspace itself varies with the realization, a point Chapter 8 takes up. For every realization,

$$\dim \mathcal{C}_\Gamma \;\le\; \operatorname{rank} \mathcal{C}_F \;\le\; \dim \mathcal{C}_\Lambda \;\le\; n. \tag{6.1}$$

Example 6.2 (The diamond)**.** Take the pattern of Fig. 6.1: input at node 1, two parallel paths 1→2→4 and 1→3→4. Symbolically,

$$\mathcal{C}_F = \big[\, e_1,\;\; a_{21}e_2 + a_{31}e_3,\;\; (a_{42}a_{21} + a_{43}a_{31})\, e_4,\;\; 0 \,\big].$$

Generically the three non-zero columns are independent: $\operatorname{rank} \mathcal{C}_F = 3$, so $\dim \mathcal{C}_\Lambda = 3$. But the only non-zero entry of the third column is a *multi-term* in the sense of Definition 4.26 (the two paths to node 4 have equal length), and the realization $a_{21} = a_{31} = a_{42} = 1$, $a_{43} = -1$ cancels it, leaving rank 2. No realization does worse, so $\dim \mathcal{C}_\Gamma = 2$. Each inequality of Eq. (6.1) is strict for some realization:

$$2 \;\le\; \operatorname{rank} \mathcal{C}_F \;\le\; 3 \;<\; 4.$$

The pattern is never controllable ($\dim \mathcal{C}_\Lambda < 4$), yet *how much* of it is controllable genuinely depends on the weights. That is exactly the information the two subspaces record. Note also what never varies: nodes 2 and 3 are controllable only in the combination $a_{21}e_2 + a_{31}e_3$, one

direction for two nodes. This is the twin-star failure of Example 5.5 appearing inside a larger graph.

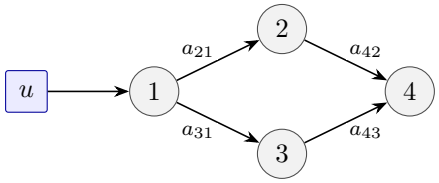


Figure 6.1: The diamond of Example 6.2: two equal-length paths to node 4 make the one non-zero entry of its row of $\mathcal{C}_F$ a multi-term, which the realization of that example makes zero, and one input for the twins $2, 3$ caps the generic dimension at three.

6.2 Algebraic Necessary and Sufficient Conditions

Theorem 6.3 (Full dimension)**.** *The family $\mathcal{Q}(A)$ with input matrix B is SC iff* $\dim \mathcal{C}_\Lambda = n$*, and SSC iff* $\dim \mathcal{C}_\Gamma = n$*.*

Both halves are read off Definition 6.1: $\dim \mathcal{C}_\Lambda = n$ says some realization has $\operatorname{rank} \mathcal{C}_F = n$, and by Lemma A.1 almost all then do. $\dim \mathcal{C}_\Gamma = n$ says the minimum over the family is n, that is, every realization is controllable.

Every graph-theoretic criterion in this book is a combinatorial condition equivalent to one of these two equalities. Revisiting Chapter 5's examples: the chain has $\dim \mathcal{C}_\Gamma = n$ (SSC), the flip family has $\dim \mathcal{C}_\Lambda = n$ but $\dim \mathcal{C}_\Gamma = n-1$ (SC only), the twin star has $\dim \mathcal{C}_\Lambda = 2 < 3$ (never), and the diamond sits strictly inside both gaps.

6.3 Generic Dimension on Graphs

Theorem 6.4 (Generic dimension)**.** $\dim \mathcal{C}_\Lambda$ *equals the maximum number of state nodes that can be covered by a disjoint set of stems and cycles on the* reachable part *(Definition 4.13) of the pattern graph.*

This is Section 5.3 made quantitative: the largest surviving minor of $\mathcal{C}_F$ is built from the largest disjoint stem-and-cycle cover, and the count is computable in polynomial time. The proof is Section A.2. The restriction to the reachable part is necessary: a covered but *unreachable* cycle establishes nothing, and dropping the restriction makes the count wrong, a case that Example 7.4 in Chapter 7 works out in full. (All three patterns below are input-connected, so there the restriction has no effect.)

Example 6.5 (Covers yield dimensions)**.** *Four-node network* (Fig. 4.1): the single stem $u{\to}1{\to}2{\to}3{\to}4$ covers all four nodes, so $\dim \mathcal{C}_\Lambda = 4$, the graph-side counterpart of the determinant $a_{21}^3 a_{32}^2 a_{43}$ computed in Example 4.25. *Diamond* (Fig. 6.1): a stem can take only one of the two paths ($u{\to}1{\to}2{\to}4$ covers $\{1, 2, 4\}$), and no disjoint cycle exists to pick up node 3, so $\dim \mathcal{C}_\Lambda = 3$, matching the symbolic computation. *Twin star*: the longest stem covers $\{1, 2\}$ or $\{1, 3\}$, so $\dim \mathcal{C}_\Lambda = 2$. In all three cases the algebra of Chapter 5 and a cover found from the graph alone agree exactly.

6.4 Bounding Worst-Case Dimension

For the worst-case dimension no such clean formula is available. This book proves one lower bound and quotes two further results from the sources listed at the end of the chapter.

The bound proved here counts the state nodes covered by disjoint *shortest* stems: stems that reach each of their nodes at its input-distance, and by the only walk of that length, so that every entry they produce is a single-term (Theorem B.1). On the diamond, the shortest stem $u{\to}1{\to}2$ has single-term entries that no realization can cancel, ensuring $\dim \mathcal{C}_\Gamma \geq 2$, which the cancellation of Example 6.2 shows is tight. With several drivers the count over disjoint shortest stems needs two further hypotheses: that no two stems reach the same node in the same number of steps, and that each stem is shortest against *all* the drivers (the input-distance of Definition 4.8), not merely against its own. Otherwise the single-terms of two drivers can cancel against each other. The statement is Theorem B.1, proved in Section B.1.

Two results are quoted here, not proved. The first concerns the undirected, free-diagonal networks of Remark 9.9, the regime in which the propagation of Section 7.4 (color the driver nodes black, then let black spread one node at a time) decides SSC exactly. There the size of the derived set (Definition 7.13) left when propagation stops is itself a lower bound on $\dim \mathcal{C}_\Gamma$. And on a hierarchical DAG (Definition 4.21) the worst-case dimension is computed exactly, layer by layer down the distance layers: the diamond is such a pattern and the four-node network of Fig. 4.1 is not (Example 4.22), so that computation applies to the first and not the second. Chapter 8 draws on the second of the two quoted results where it needs a computable $\dim \mathcal{C}_\Gamma$.

6.5 The Four Controllable Subspaces

Two more subspaces complete the picture. Beyond its *dimension*, the controllable subspace also changes *orientation* as the parameters move. Intersecting it over a family of realizations yields the *fixed* subspaces: the fixed structurally controllable subspace (FSCS), which intersects over the generic-dimension realizations, and the fixed strongly structurally controllable subspace (FSSCS), which intersects over *all* realizations (Definition 8.2). Each collects the directions controllable throughout the family it runs over. The generic-dimension realizations are a subset of the whole family, so $\mathcal{C}_\Gamma^F \subseteq \mathcal{C}_\Lambda^F$. Figure 6.2 shows the containment relations among the four. The fixed pair and its node-level meaning are the subject of Chapter 8.

6.6 Grid: Subspace Row

	SC (almost all)	**SSC** (all)
algebra	$\dim \mathcal{C}_\Lambda = n$	$\dim \mathcal{C}_\Gamma = n$
graph	stems+cycles cover count (Theorem 6.4)	bounds; exact only on HDAGs

Both columns of this row still rely on algebra: a dimension to compute, bounds to establish. The next chapter asks whether SC and SSC of the whole network can instead be decided from the graph alone (Chapter 7).

Sources. The generic dimension as a cover count is Hosoe [19], resting on the generic-rank analysis of [18]. The four controllable subspaces of Section 6.5, Fig. 6.2 and the exact worst-case

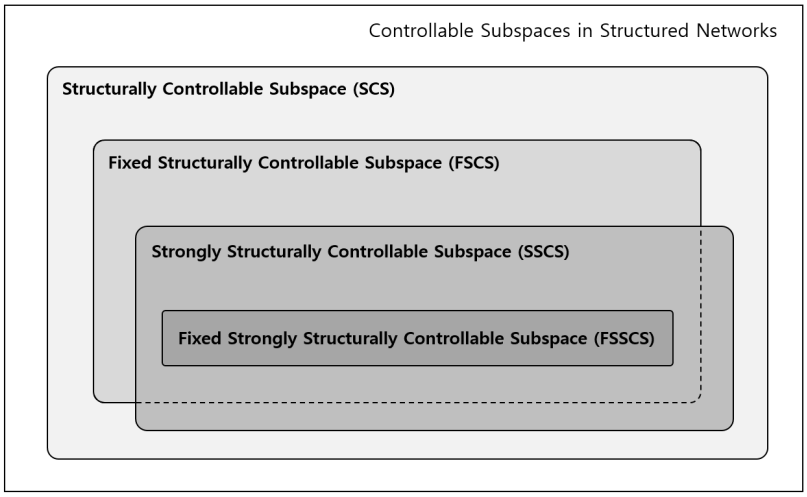


Figure 6.2: The four controllable subspaces of a structured network and the inclusions among them (figure from the author's dissertation [1], and the four-subspace framework was introduced in [12]). Each fixed subspace lies inside the controllable subspace of every realization it is intersected over (the FSCS inside every generic one, the FSSCS inside *every* realization), and $\dim \mathcal{C}_\Gamma \le \dim \mathcal{C}_\Lambda$. The innermost containment $\mathcal{C}_\Gamma^F \subseteq \mathcal{C}_\Lambda^F$ holds because the FSSCS intersects over the larger family (Definition 8.2 and Remark 8.5). The outer two regions compare *dimensions*, not subspaces: $\mathcal{C}_\Lambda$ and $\mathcal{C}_\Gamma$ each denote the controllable subspace of one realization attaining the maximum (respectively minimum) rank, and that subspace moves with the realization (Definition 6.1). This chapter concerned the outer pair SCS/SSCS, and Chapter 8 enters the fixed pair.

dimension on hierarchical DAGs are the author's [12], with the unified treatment in [1]. The derived-set lower bound quoted in Section 6.4 is [20], whose distance-based companion is [21]. The survey [4] collects both.

Exercises

Exercise 6.1 (What the pattern fixes)**.** Say which of $\dim \mathcal{C}_\Lambda$, $\mathcal{C}_\Lambda$, $\dim \mathcal{C}_\Gamma$ and $\mathcal{C}_\Gamma$ are invariants of the pattern and which move with the realization (Definition 6.1). Then write out Eq. (6.1) for the diamond of Example 6.2 and say which of its inequalities is strict no matter which realization is taken.

Exercise 6.2 (Cover counts)**.** Use Theorem 6.4 to compute $\dim \mathcal{C}_\Lambda$ for (a) the fan of Example 4.29 and (b) the diamond of Example 6.2 with one further edge $4 \to 5$ appended. Name in each case a disjoint set of stems and cycles attaining the count, and check both numbers against the matrices $\mathcal{C}_F$ of Section 4.4.3.

Exercise 6.3 (A covered cycle that counts for nothing)**.** Take the four-node pattern with driver node 1, the edge $1 \to 2$, and the two edges $3 \to 4$, $4 \to 3$. Count the state nodes covered by a disjoint set of stems and cycles on the whole graph, then compute $\mathcal{C}_F$ and the maximum rank it attains over the family. Explain the difference by the restriction of Theorem 6.4 to the reachable part.

Exercise 6.4 (Shortest stems)**.** For the fan and for the diamond with the edge $4 \to 5$ appended, find a longest stem that reaches each of its nodes at that node's input-distance and by the

only walk of that length (Section 6.4), and state the lower bound on $\dim \mathcal{C}_\Gamma$ it gives. Exhibit for each pattern a realization at which the bound is attained. [Node 5 of the fan and node 4 of the diamond are each reached by more than one walk of the same length.]

Exercise 6.5 (One direction for two nodes)**.** In the diamond of Example 6.2, show that no column of $\mathcal{C}_F$ other than the second has a non-zero entry in rows 2 and 3, and give the one direction of $\operatorname{span}\{e_2, e_3\}$ that lies in the controllable subspace. Explain why nodes 2 and 3 are never independently controllable, and why that direction is nevertheless different at different realizations.

Exercise 6.6 (Four subspaces)**.** Decide true or false, with a reason from Section 6.5: (a) the FSSCS is the intersection of the controllable subspaces of all realizations of the family, (b) $\mathcal{C}_\Gamma^F \subseteq \mathcal{C}_\Lambda^F$, and (c) $\mathcal{C}_\Lambda$ and $\mathcal{C}_\Gamma$ are the controllable subspaces of one and the same realization.

Exercise 6.7 (Full dimension)**.** Give the pair $(\dim \mathcal{C}_\Lambda, \dim \mathcal{C}_\Gamma)$ for the fan, for the diamond with the edge 4→5 appended, and for the three-node pattern with driver node 1 and edges 1→2, 2→3, 1→3. Then use Theorem 6.3 to decide in each case whether the family is SC, SSC, or neither.

Chapter 7

Whole-Graph Criteria

The graph-theoretic necessary and sufficient conditions for SC and SSC of the entire network: Lin's stems-and-cycles criterion as the combinatorial condition equivalent to $\dim \mathcal{C}_\Lambda = n$*, and Mayeda's two subset conditions as the condition equivalent to* $\dim \mathcal{C}_\Gamma = n$*. The needed vocabulary (matchings for the SC side, dedicated nodes and zero forcing for the SSC side) is introduced on the spot, and every criterion is stated back-to-back with its algebraic counterpart. The most-quoted versions of these theorems carry hypotheses that are often dropped (input-connectedness for Lin, free self-loops for the zero-forcing test), and each of the two comes with a hand-checkable counterexample showing what goes wrong when it is dropped. One of the two counterexamples is our running four-node network.*

7.1 Structural Controllability of Graphs

Theorem 6.3 left one task: establish $\dim \mathcal{C}_\Lambda = n$ from the graph alone. By Theorem 6.4 the generic dimension is a cover count, so the whole-graph question reduces to whether a disjoint set of stems and cycles can cover *all* state nodes. Almost: the count of Theorem 6.4 is computed on the *reachable* part of the pattern graph, and when the criterion is quoted as a bare cover condition that hypothesis is silently lost. We state it with that hypothesis, in the two classical forms.

Definition 7.1 (Dilation)**.** For a set $S \subseteq \mathcal{V}^S$ of state nodes, its in-neighbor set $N^{\text{in}}(S)$ (Definition 4.5) collects every node (state or input) with at least one edge into S. The pattern graph has a *dilation* if $|N^{\text{in}}(S)| < |S|$ for some non-empty S: more nodes than in-neighbors to feed them. Such a set S is itself called a *dilation*, and the pattern graph is *dilation-free* if it has none.

Theorem 7.2 (Structural controllability of a graph)**.** *For a pattern graph with at least one input, the following are equivalent:*

(i) the family is SC.

(ii) the pattern is input-connected (Definition 4.13), and some disjoint set of stems and cycles covers all state nodes.

(iii) the pattern is input-connected, and no dilation exists.

Both graph conditions are checkable in polynomial time [18], and each direction of failure is visible in two lines. If some state node is unreachable, no walk from any input reaches it, so its row of $\mathcal{C}_F$ is identically zero (Proposition 4.24). If S is a dilation, the rows of S in $[A_F\ B]$ have their non-zeros only in the columns of $N^{\text{in}}(S)$, so those $|S|$ rows have rank at most $|N^{\text{in}}(S)| < |S|$ *for every realization*: the PBH test (Theorem 3.8) always fails at $\lambda = 0$. Note what this means against the trichotomy of Proposition 5.4: a whole-graph SC failure admits no exception. If the family is not SC, no realization at all is controllable. Sufficiency is Lin's theorem [16, 22]. The classical proof rebuilds the graph as a union of *cacti* (stems carrying cycle *buds*, each bud joined to the stem by a single *bridge edge*), and a spanning family of cacti is precisely condition (ii). That construction, and the equivalence (ii)⇔(iii), are carried out in Section A.3.

Example 7.3 (Three graphs decided by Lin's theorem)**.** *Four-node network* (Fig. 4.1): all four nodes are reachable (Example 4.15) and the single stem $u{\to}1{\to}2{\to}3{\to}4$ is a cover, so the family is SC by Theorem 7.2, the graph-side reading of $\det \mathcal{C}_F = a_{21}^3 a_{32}^2 a_{43} \not\equiv 0$. *Diamond* (Fig. 6.1): input-connected, but $S = \{2, 3\}$ has $N^{\text{in}}(S) = \{1\}$, a dilation, and equivalently no cover exceeds three nodes (Example 6.5), so the family is not SC. *Twin star* (Example 3.9): the same dilation, $S = \{2, 3\}$ with $N^{\text{in}}(S) = \{1\}$, which is the graph reading of $\det \mathcal{C}_F \equiv 0$ in Example 5.5. The diamond and the star show that input-connectedness alone proves nothing. Figure 7.1 shows the converse failure.

Example 7.4 (Covered but not reachable from the inputs)**.** In Fig. 7.1, the stem $u{\to}1$ and the cycle $2{\to}3{\to}2$ are disjoint and together cover all three state nodes, yet nodes $2, 3$ are unreachable from the input, and $\dim \mathcal{C}_\Lambda = 1$. A covered cycle only recycles what already arrives from a stem. Feed it nothing and it yields nothing. This is why input-connectedness must be stated as a hypothesis and why the cover count of Theorem 6.4 is taken on the reachable part.

Figure 7.1: Why input-connectedness is needed (Example 7.4): stem $u{\to}1$ plus cycle $2{\to}3{\to}2$ cover every state node, but the cycle is unreachable and the family is far from SC.

7.2 Matchings

The dilation condition has an equivalent form in terms of matchings, which organizes the entire SC side of Part III. We fix the vocabulary now.

Definition 7.5 (Matching)**.** For an edge $j \to i$, node j is its *tail* and node i its *head*. A *matching* is a set of edges no two of which share a tail or a head. A state node is *matched* (by a given matching) if it is the head of one of the matching's edges, and *unmatched* otherwise. A matching of largest possible size is a *maximum matching*, and one that matches every state node is *perfect*.

A matching is a system of node-disjoint paths and cycles (follow matched edges tail-to-head), which is exactly the shape of the covers in Theorem 7.2. The precise statement is Hall's marriage theorem applied to the bipartite graph representation with tails on one side and state nodes on the other (see, *e.g.*, [7]): *some matching that may use input edges matches every state*

node if and only if the pattern graph is dilation-free. Counting instead of deciding gives the result of Liu et al. [3]: with a maximum matching taken on the *state subgraph*, the subgraph induced by the state nodes (Definition 4.11) with input edges dropped, the minimum number of inputs for SC equals the number of unmatched nodes, and is one when the matching is perfect. That count, its non-uniqueness, the hypotheses it carries, and where the inputs must attach are the subject of Chapter 9. Here matchings serve as vocabulary.

Example 7.6 (Matchings of the four-node network)**.** On the state subgraph of Fig. 4.1, the edge set $\{1{\to}2,\ 2{\to}3,\ 3{\to}4\}$ is a matching: tails $1, 2, 3$ distinct, heads $2, 3, 4$ distinct. It matches $2, 3, 4$ and leaves node 1 unmatched, precisely the driver node, which is why one input suffices. No matching of size four exists, so this matching is maximum. Indeed, a perfect matching would need four distinct heads, but only $3{\to}1$ can match node 1, consuming tail 3. Heads 3 and 4 must then share the single remaining tail 2, via $2{\to}3$ and $2{\to}4$, so one stays unmatched. Maximum matchings are not unique: $\{1{\to}2,\ 2{\to}3,\ 3{\to}1\}$ also has size three but leaves node 4 unmatched instead. Different maximum matchings expose different nodes (same count, different placement), a fact Chapter 9 returns to. For the diamond, every matching must choose between the heads 2 and 3 (both have the single tail 1): a maximum matching such as $\{1{\to}2,\ 2{\to}4\}$ leaves *two* nodes unmatched, the matching form of its dilation.

7.3 Strong Structural Controllability of Graphs

For SSC the condition must hold for all realizations, so no cancellation may be possible at any of them, and the cost is a condition on every *subset* of state nodes, not on one cover. The subset vocabulary comes from the author's reinterpretation of Mayeda's theorem [1, 23]:

Definition 7.7 (Dedicated and sharing nodes)**.** Let $S \subseteq \mathcal{V}^S$ be non-empty. A node v (state or input) with *exactly one* edge into S is a *dedicated node* of S, and a node with two or more edges into S is a *sharing node* of S.

A dedicated node touches S through a single weight, and a single non-zero weight cannot be canceled. One linear-algebra lemma makes this precise. Recall from Definition 4.1 that a *pattern matrix* is a matrix each of whose entries is either structurally zero or a free non-zero parameter.

Lemma 7.8 (Full row rank for all realizations)**.** *A pattern matrix has full row rank at every realization if and only if every non-empty set S of rows admits a column with exactly one non-zero entry in S.*

Proof. For the *if* condition, assume every non-empty set of rows admits such a column. For contradiction, suppose that some realization has a linear dependency among its rows. Let S be the support of one. The column with exactly one non-zero in S contributes a single term (non-zero coefficient times non-zero entry) to the dependency. That is impossible. For the *only if* condition, we prove the contrapositive: suppose that some S has no such column. Give every row of S the coefficient 1 and choose each column's free entries within S to sum to zero, which is possible since each column has none or at least two of them. The result is a realization with dependent rows. □

Now run the PBH test (Theorem 3.8) over the whole family, splitting on λ. At $\lambda = 0$ the test matrix is $[A_F\ B]$. Its column v has non-zeros exactly in the rows that v feeds, so "a

column with exactly one non-zero in the rows S" *is* a dedicated node of S, possibly a member of S itself, whose single edge into S may be an out-edge to another member, or, where the pattern carries one, its own self-loop. At $\lambda \neq 0$ the shift $-\lambda$ puts a non-vanishing entry on every diagonal position, and two things change. (a) If a node $s \in S$ has no out-neighbor inside S, its column meets the rows S only in the diagonal entry $-\lambda$, so no dependency can have support all of S: (C2) asks nothing of such subsets. (b) A member of S with exactly one edge into S is no longer dedicated, since the shift puts a second non-zero, $-\lambda$, in its own row, so its single edge weight can be balanced against that diagonal entry. What survives is exactly Mayeda's pair of conditions [17], in modern algebraic form [24]:

Theorem 7.9 (Strong structural controllability of a graph)**.** *Let each input attach to a single driver node (the standing convention of Part II, declared in Section 4.1). The family is SSC if and only if both hold:*

(C1) every non-empty $S \subseteq \mathcal{V}^S$ has a dedicated node in $N^{\text{in}}(S)$, possibly a node of S itself.

(C2) every non-empty $S \subseteq \mathcal{V}^S$ with $S \subseteq N^{\text{in}}(S)$ has a dedicated node in $N^{\text{in}}(S) \setminus S$.

Neither condition implies the other, and both are needed: (C1) alone is the $\lambda = 0$ test, (C2) alone the $\lambda \neq 0$ test, and the proof splitting on λ is Section B.2. Note also the relation to Definition 7.1: a dilation means S has too few in-neighbors, while failing (C1) means S has no *dedicated* in-neighbor. The in-neighbors may be many and all sharing. Dedicated-node analysis is dilation analysis sharpened from counting in-neighbors to counting each in-neighbor's edges into S [1]. The cost of the sharpening is that the conditions quantify over all 2^n subsets, so a direct check is exponential in n (see Remark 7.17 for propagation criteria that avoid the enumeration). For patterns with a self-loop at every state node the two conditions collapse into one: (C2) absorbs (C1) [23]. The next section gives a test of polynomial complexity in exactly that regime (Theorem 7.14).

Example 7.10 (The four-node network passes both conditions)**.** For the pattern of Fig. 4.1, every subset has its dedicated node, and the interesting cases are:

- $S = \{3, 4\}$: node 2 is *sharing* (edges into both 3 and 4), but node $3 \in S$ is dedicated. Its only edge into S is $3 \to 4$, since $3 \to 1$ leaves S. (C1) holds through a member of S, and node 4 has no out-neighbor in S, so (C2) asks nothing of this subset.
- $S = \{1, 2, 3\}$: a cycle, in which every node has an out-neighbor inside, so (C2) demands an outside dedicated node, and the input provides it: u has exactly one edge into S, at node 1.
- $S = \mathcal{V}^S$: again u is dedicated. Since node 4 has no out-edges, (C2) asks nothing of any subset containing it.

The remaining subsets are routine, and both conditions hold: the family is SSC, the subset-level reading of the single-monomial determinant $a_{21}^3 a_{32}^2 a_{43}$ of Example 4.25.

Example 7.11 (Diamond and star fail (C1))**.** For the diamond (Fig. 6.1), take $S = \{2, 3\}$: node 1 is a sharing node (edges into 2 and 3), the input reaches only 1, and nodes $2, 3$ send nothing into S, so no dedicated node exists and (C1) fails, the same subset that already carried the dilation. The twin star fails identically at its two end nodes. Both families are not SSC for the strongest reason: they are not even SC.

Remark 7.12 (A sharing input: the flip family). The flip family (Example 5.3) is SC but not SSC, and Definition 7.7 explains why even though its input $b = (1, 1)^\top$ feeds two nodes, outside

the single-attachment convention of Theorem 7.9: for $S = \{1, 2\}$ every in-neighbor (node 1, node 2, and the input itself) feeds both nodes of S. All sharing, no dedicated node, and indeed the balanced realizations $a_{21} + a_{22} = a_{11} + a_{12}$ are uncontrollable.

7.4 Zero Forcing

The subset conditions of Theorem 7.9 quantify over all 2^n subsets. *Zero forcing* replaces that enumeration with a propagation process that reads the graph alone [25, 26].

Definition 7.13 (Zero forcing)**.** Color the driver nodes black and all other state nodes white, then apply the *color-change rule* until no more white nodes can be colored black: a black node with exactly one white out-neighbor forces that neighbor black. The final set of black nodes is the *derived set*. The driver set (the driver nodes taken together, the design variable of Problem 9.1) is a *zero forcing set* (ZFS) if the derived set is all of $\mathcal{V}^S$.

Forcing never uses a weight: each step counts the white out-neighbors of one black node, and the process is over as soon as no black node has exactly one. Such a node is, in the vocabulary of Definition 7.7, a dedicated node of the current white set lying outside it, so forcing searches for outside dedicated nodes one step at a time instead of one subset at a time. When every diagonal entry is free that search is all (C1) and (C2) ask for, and the test is exact:

Theorem 7.14 (ZFS criterion)**.** *Suppose every diagonal entry of A_F is a free parameter (a self-loop of arbitrary weight, zero included, permitted at every state node) while off-diagonal entries follow the pattern graph as usual. Then the family is SSC if and only if the driver set is a zero forcing set. The test is polynomial time.*

The hypothesis cannot be dropped. With free diagonals the two ways in which a loop-free pattern can satisfy (C1)/(C2) *without an outside dedicated node* both close. A member of S can no longer serve as its dedicated node: if it has an edge into S, its own loop makes two, and if its only entry in S is that loop, the weight is free *including zero*, so it cannot be the guaranteed single non-zero that Lemma 7.8 asks for. And a node without out-neighbors in S now has one, its own loop, so (C2) applies to its subsets as well. Mayeda's two conditions therefore collapse into the single demand "every S has a dedicated node outside S", which is precisely what forcing checks. The collapse is proved in Section B.3. Drop the hypothesis and the equivalence breaks, in exactly one direction: forcing remains sufficient for SSC (the free-diagonal family contains the zero-diagonal one), but it is no longer necessary. Our four-node network is the counterexample.

Example 7.15 (The four-node network: SSC without a ZFS)**.** Run forcing on Fig. 7.2. The driver 1 forces 2 (its only out-neighbor). Then node 2 has *two* white out-neighbors, 3 and 4, and the process stops with derived set $\{1, 2\}$. The drivers are not a ZFS, yet the family is SSC (Example 7.10). The remaining white set $\{3, 4\}$ carries the two cases that forcing cannot check: its dedicated node is node 3, *inside* that set, and the subsets containing node 4 are left unconstrained by (C2), whose λ-shift asks nothing of a subset containing a node with no out-neighbor inside it. Neither case is available to a rule that only lets outside black nodes act. The conclusion of Theorem 7.14 is nonetheless correct *for its own hypothesis*: allow self-loops and controllability fails at some realization. With unit edge weights and a unit self-loop at

node 3, rows 3 and 4 of the realization

$$A' = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

are identical, so $z = e_3 - e_4$ satisfies $z^\top A' = 0$ and $z^\top b = 0$: PBH fails at $\lambda = 0$. Symbolically, self-loops d_3, d_4 at nodes $3, 4$ give

$$\det \mathcal{C}_F = a_{21}^3\, a_{32}(a_{32}a_{43} + a_{42}(d_4 - d_3)).$$

This is a sum of monomials that vanishes at some realization, whereas the zero-diagonal family keeps a single monomial, which no realization can cancel. In short: state the zero-forcing criterion *with* its self-loop hypothesis. For structurally loop-free patterns it is a sufficient condition only, and the exact criterion remains Theorem 7.9.

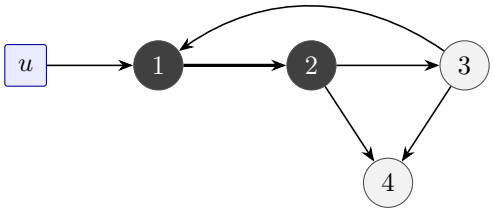


Figure 7.2: Forcing on the four-node pattern (Example 7.15): 1 forces 2, then node 2 has two white out-neighbors (3 and 4) and the process stops with derived set $\{1, 2\}$, although the family is SSC.

Example 7.16 (Forcing on the chain)**.** For the chain $u \to 1 \to 2 \to 3$ of Example 5.7, forcing proceeds along the chain: 1 forces 2, and 2 forces 3. The driver set is therefore a ZFS, hence SSC even with arbitrary self-loops, and *a fortiori* for the loop-free chain, matching $\det \mathcal{C}_F = a_{21}^2 a_{32}$. The order in which the nodes are forced (the *forcing chain*) retraces the stem. That is no accident: a ZFS corresponds to a *constrained matching* (a matching that is the unique one on its node set) whose paths are exactly the forcing chains [26]. Uniqueness is the matching-side form of "nothing can cancel".

Remark 7.17 (Repairing the criterion for loop-free patterns). The gap shown by Example 7.15 can be closed: for patterns with structurally zero diagonals, exact zero-forcing-type criteria have been developed that run *two* propagation rules on the graph with its self-loops *specified*, one rule per Mayeda condition, covering both a dedicated node inside S and the subsets that (C2) does not constrain. Constrained matchings give the equivalent combinatorial form of the plain forcing test [26], and the survey [4] collects these variants. The minimum *size* of a zero forcing set, and its role as the SSC counterpart of Liu's driver count, belongs to Chapter 9.

7.5 Grid: Whole-Graph Row

The whole-graph row of the book's running grid is now complete. Each graph cell is the exact combinatorial necessary and sufficient condition for the algebraic cell above it.

	SC (almost all)	**SSC** (all)
algebra	$\dim \mathcal{C}_\Lambda = n$ (Theorem 6.3)	$\dim \mathcal{C}_\Gamma = n$ (Theorem 6.3)
graph	input-connected + stems/cycles cover = no dilation (Theorem 7.2)	dedicated node for every subset (C1)∧(C2) (Theorem 7.9)
cost	polynomial [18]	direct check exponential; ZFS polynomial under free self-loops (Theorem 7.14)

Remark 7.18 (Symmetry, dilation, and the PBH picture). Every whole-graph failure in this chapter is a PBH failure at some mode, which the graph exhibits. In the twin star, *every* realization admits the left null vector $z = (0, a_{31}, -a_{21})$: $z^\top A_F = 0$ and $z^\top b = 0$, the left eigenvector of Theorem 3.8 concentrated on the dilation $\{2, 3\}$. At equal weights it becomes the antisymmetric direction $e_2 - e_3$ of the end-node-swapping automorphism that fixes the driver. Equitable partitions systematize this: nodes grouped symmetrically with respect to the drivers generate uncontrollable modes, giving graph-symmetry *necessary* conditions for SC [27, 7]. Dilations, sharing nodes, remaining white sets, and quotient symmetries are four forms of one statement: some direction of state space is fed by too few independent inputs.

So the whole graph can be decided from its structure alone: covers for SC, dedicated nodes for SSC, forcing when self-loops are free. But most real networks simply fail these tests, and a conclusion of "not controllable" for the whole graph is where analysis starts, not ends: which individual nodes remain controllable at every realization? That is the question of Chapter 8.

Sources. Lin's theorem and its formulations are [16, 22, 19, 18]. Matchings and the driver count follow [3, 7]. Mayeda's two conditions and their modern form are [17, 24], and the dedicated/sharing-node reinterpretation and the self-loop collapse are [1, 23]. Zero forcing and constrained matchings follow [25, 26, 28], and the symmetry conditions are [27]. The survey [4] treats the whole chapter.

Exercises

Exercise 7.1 (Lin's three conditions)**.** State the three equivalent conditions of Theorem 7.2. Then explain, in two or three sentences, why input-connectedness cannot be dropped from condition (ii), using the pattern of Fig. 7.1.

Exercise 7.2 (A dilation in the fan)**.** Apply Theorem 7.2 to the fan of Example 4.29. Exhibit a set S that is a dilation (Definition 7.1) together with its set $N^{\text{in}}(S)$, decide whether the family is SC, and give $\dim \mathcal{C}_\Lambda$ by Theorem 6.4.

Exercise 7.3 (All maximum matchings)**.** On the state subgraph of Fig. 4.1, list *all* maximum matchings (Definition 7.5) and the node each one leaves unmatched. Why can no matching there have size four?

Exercise 7.4 (Repairing an unreachable cycle)**.** Add the edge $1 \to 2$ to the pattern of Fig. 7.1, keeping $u \to 1$ and the cycle $2 \to 3 \to 2$. Decide SC by Theorem 7.2, then decide SSC by Theorem 7.9: list the subsets on which (C2) imposes a condition and name an outside dedicated node for each. Confirm the answer by computing $\det \mathcal{C}_F$.

Exercise 7.5 (Where (C2) imposes a condition)**.** Example 7.10 checks three subsets and calls the rest routine. Complete the (C2) half: list every non-empty $S \subseteq \mathcal{V}^S$ of the four-node

network on which (C2) imposes a condition, and name its outside dedicated node. [Hint: node 4 has no out-edges.] Then delete the edge 3→1 and repeat: which subsets does (C2) constrain now, and which feature of the modified pattern accounts for the answer?

Exercise 7.6 (Dedicated nodes sharpen dilations)**.** Explain, in three or four sentences, how the dedicated-node condition (C1) sharpens the dilation condition of Definition 7.1, and give a subset of the diamond on which both fail.

Exercise 7.7 (Forcing on the diamond)**.** Color node 1 of the diamond (Fig. 6.1) black and apply the color-change rule of Definition 7.13. What is the derived set? Then decide which of $\{1,2\}$, $\{1,4\}$, $\{2,3\}$ are zero forcing sets, giving the derived set in each case.

Exercise 7.8 (Zero forcing sets of the fan)**.** For the fan of Example 4.29, show that $\{1,2,3\}$ is a zero forcing set and that no two-node driver set is. [Hint: node 1 has no state in-neighbor, so it can never be forced.]

Exercise 7.9 (Forcing with free self-loops)**.** Take the four-agent graph of Example 2.10 (undirected edges 12, 23, 34, 13, each read as two opposite directed edges) and give every state node a self-loop of free weight, so that Theorem 7.14 applies. Decide which of the driver sets $\{4\}$, $\{3,4\}$, $\{1,4\}$ make the family SSC.

Exercise 7.10 (What forcing cannot check)**.** Explain why Theorem 7.14 gives only a sufficient condition for SSC on a structurally loop-free pattern. Then delete the edge 3→1 from the four-node network (Fig. 4.1), keeping the driver node 1: give the derived set, establish SSC by computing $\det \mathcal{C}_F$, and name the two situations in the remaining white set that the color-change rule cannot check.

Chapter 8

Node-Level Controllability: Fixed Subspaces and Fixed Nodes

Most real networks fail the whole-graph tests of the previous chapter, so the practical question is which individual *nodes remain controllable at every realization. Even at fixed dimension the controllable subspace* rotates *as the parameters move. The directions that survive every rotation form the fixed subspaces FSCS and FSSCS, and the coordinate axes they contain name the fixed nodes. Both notions come, as always, in an SC and an SSC version, each with an algebraic comparison (add an input, watch a dimension) and a graph criterion: matched layers on hierarchical DAGs for the SC side, shortest control paths for the SSC side. The SC side comes out exact. On the SSC side the algebraic comparison decides in one direction only, and the graph criterion is exact for a related class of nodes. A ranking of the nodes by robustness to weight uncertainty completes the book's summary grid.*

8.1 Orientation Problem

Chapter 6 settled the *dimension* of the controllable subspace: it moves between $\dim \mathcal{C}_\Gamma$ and $\dim \mathcal{C}_\Lambda$, both invariants of the pattern. But a subspace is more than its dimension. Two realizations may agree that the controllable subspace is a plane and still disagree about *which* plane, and a state controllable in one of them may be uncontrollable in the other. The following three-node pattern shows this in its simplest form.

Example 8.1 (A rotating plane)**.** Drive node 1 and let it feed two nodes that also feed each other (Fig. 8.1): $u \to 1$, edges $1 \to 2$ (a_{21}), $1 \to 3$ (a_{31}), $2 \to 3$ (a_{32}), $3 \to 2$ (a_{23}). By stem enumeration (Proposition 4.24),

$$\mathcal{C}_F = \begin{bmatrix} 1 & 0 & 0 \\ 0 & a_{21} & a_{23}a_{31} \\ 0 & a_{31} & a_{32}a_{21} \end{bmatrix}, \qquad \det \mathcal{C}_F = a_{21}^2 a_{32} - a_{23}a_{31}^2.$$

Each non-zero entry is a *single-term* (Definition 4.26), one WP with one walk per length and target: $a_{23}a_{31}$ is $u{\to}1{\to}3{\to}2$, $a_{32}a_{21}$ is $u{\to}1{\to}2{\to}3$. The cancellation therefore happens not inside an entry, as in the diamond, but across the 2×2 minor. Generically the determinant is non-zero, so the pattern is SC and $\dim \mathcal{C}_\Lambda = 3$. On the variety $a_{21}^2 a_{32} = a_{23}a_{31}^2$ the rank drops

to 2, and never lower (the second column cannot vanish), so $\dim \mathcal{C}_\Gamma = 2$. On that variety the third column is proportional to the second, and the controllable subspace is the plane

$$\operatorname{Im} \mathcal{C}_F = \operatorname{span}\{\, e_1,\ a_{21}e_2 + a_{31}e_3 \,\}.$$

Every such plane contains the x_1-axis, and as the ratio $a_{21} : a_{31}$ sweeps the variety the plane *rotates* about that axis: Fig. 8.2 shows three worst-case realizations, from nearly the x_1x_2-plane to nearly the x_1x_3-plane. The dimension is pinned at two, yet whether x_2 (or x_3) is controllable depends entirely on the realization. Only the axis on which all the planes agree (the x_1-axis) is controllable in every worst case.

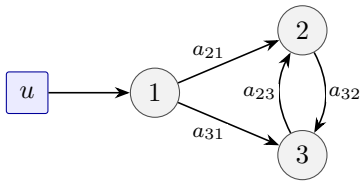
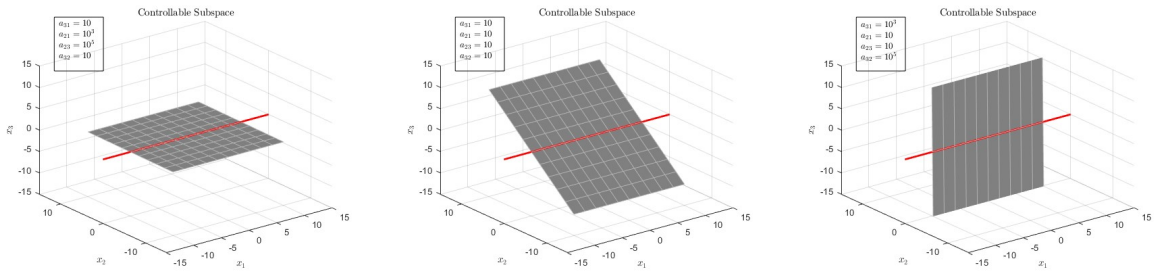


Figure 8.1: The pattern of Example 8.1: one driver, two mutually coupled followers. SC but not SSC, with worst-case rank 2 on the variety $a_{21}^2 a_{32} = a_{23} a_{31}^2$.



(a) $a_{21} \gg a_{31}$ (b) $a_{21} = a_{31}$ (c) $a_{21} \ll a_{31}$

Figure 8.2: Three worst-case realizations of Example 8.1, all with $a_{21}^2 a_{32} = a_{23} a_{31}^2$ (figures from the author's dissertation [1]). The controllable plane $\operatorname{span}\{e_1,\ a_{21}e_2 + a_{31}e_3\}$ rotates about the x_1-axis (red line): (a) $a_{21} = 10^3$, $a_{23} = 10^5$, $a_{31} = a_{32} = 10$ (almost the x_1x_2-plane), (b) all weights 10 (the diagonal position), and (c) $a_{31} = 10^3$, $a_{32} = 10^5$, $a_{21} = a_{23} = 10$ (almost the x_1x_3-plane). The axis common to every position is the fixed subspace.

8.2 Fixed Subspaces

The cure for a rotating subspace is an intersection: keep only the directions that every realization retains.

Definition 8.2 (FSCS and FSSCS)**.** For a pattern matrix A with family $\mathcal{Q}(A)$ and input matrix B, the *fixed structurally controllable subspace* and the *fixed strongly structurally controllable subspace* are

$$\mathcal{C}_\Lambda^F = \bigcap_{\substack{A_F \in \mathcal{Q}(A) \\ \operatorname{rank} \mathcal{C}_F = \dim \mathcal{C}_\Lambda}} \operatorname{Im} \mathcal{C}_F, \qquad \mathcal{C}_\Gamma^F = \bigcap_{A_F \in \mathcal{Q}(A)} \operatorname{Im} \mathcal{C}_F :$$

the controllable directions common to the generic-dimension realizations, and those common to *all* realizations.

The two quantifiers are those of Definitions 5.1 and 5.2: the FSCS takes the almost-all side and intersects over the generic-dimension realizations alone, the FSSCS takes the all side and intersects over the whole family. That is this book's convention throughout [29, 12], and under it an FSSC node is exactly a node controllable at every realization.

Unlike $\mathcal{C}_\Lambda$ and $\mathcal{C}_\Gamma$, which fix a dimension but admit countless realizations, the fixed subspaces are *unique*: intersections do not depend on the realization. Each lies inside the controllable subspace of every realization it is intersected over (the FSSCS inside that of *every* realization, the FSCS inside that of every generic one), and each is the largest subspace that does. Because the FSSCS intersects over a family that *contains* the generic realizations, it cannot be the larger of the two:

$$\mathcal{C}_\Gamma^F \subseteq \mathcal{C}_\Lambda^F.$$

No hypothesis is needed (intersecting more subspaces leaves less), and Remark 8.5 shows what goes wrong if the FSSCS is read over the worst-case realizations alone. The membership criteria for the two sides are collected in Section B.7.

Example 8.3 (Fixed subspaces of the rotating plane)**.** In Example 8.1 the pattern is SC, so every generic realization has $\operatorname{Im}\mathcal{C}_F = \mathbb{R}^3$ and $\mathcal{C}_\Lambda^F = \mathbb{R}^3$. On the SC side the fixed question is vacuous exactly when the whole-graph test passes. The FSSCS intersects over every realization. The generic ones contribute $\mathbb{R}^3$, and the rank-deficient ones are the rotating planes, two positions of which already leave only the common axis: $\operatorname{span}\{e_1, 10e_2+10e_3\} \cap \operatorname{span}\{e_1, 10e_2+10^3e_3\} = \operatorname{span}\{e_1\}$, so

$$\mathcal{C}_\Gamma^F = \operatorname{span}\{e_1\},$$

of dimension $1 < 2 = \dim\mathcal{C}_\Gamma$: the worst case always controls a plane, but only one *fixed* direction lies in all of them.

Example 8.4 (Fixed subspaces of the diamond)**.** The diamond (Example 6.2 and Fig. 6.1) has $\mathcal{C}_F = [\,e_1,\ a_{21}e_2+a_{31}e_3,\ (a_{42}a_{21}+a_{43}a_{31})e_4,\ 0\,]$. The generic realizations ($a_{42}a_{21}+a_{43}a_{31} \neq 0$) span $\operatorname{span}\{e_1,\ a_{21}e_2+a_{31}e_3,\ e_4\}$. The middle direction rotates with $a_{21}:a_{31}$ while e_1 and e_4 stay put, so

$$\mathcal{C}_\Lambda^F = \operatorname{span}\{e_1, e_4\},$$

of dimension $2 < 3 = \dim\mathcal{C}_\Lambda$: even the generic subspace carries a rotating direction that the intersection discards. The FSSCS intersects over all realizations, so it is cut further by the cancellation locus $a_{42}a_{21} = -a_{43}a_{31}$, where $\mathcal{C}_F$ spans only the plane $\operatorname{span}\{e_1,\ a_{21}e_2+a_{31}e_3\}$ (the rotating-plane picture again, appearing inside the diamond), leaving

$$\mathcal{C}_\Gamma^F = \operatorname{span}\{e_1\} \subsetneq \mathcal{C}_\Lambda^F.$$

Remark 8.5 (Why the intersection runs over all realizations)*.* Take the input-connected HDAG with driver 1 and state edges $1\to2$, $1\to3$, $2\to4$, $2\to5$, $3\to5$, $5\to6$ (Fig. 8.3), whose multi-terms are all multiples of $a_{21}a_{52}+a_{31}a_{53}$: generically $\dim\mathcal{C}_\Lambda = 4$, and where that term vanishes the rank drops to $\dim\mathcal{C}_\Gamma = 3$. The worst-case realizations alone intersect to $\operatorname{span}\{e_1, e_4\}$ and the generic ones to $\operatorname{span}\{e_1, e_6\}$. Neither contains the other, so reading the FSSCS as the first would leave the two fixed subspaces unordered. Definition 8.2 intersects over both instead and returns $\mathcal{C}_\Gamma^F = \operatorname{span}\{e_1\}$: node 4, controllable at every worst-case realization and at no generic one, is not an FSSC node. The source [12] states its results for the nodes controllable at every realization (the reading adopted here), though its Definition 1 is worded as the worst-case intersection.

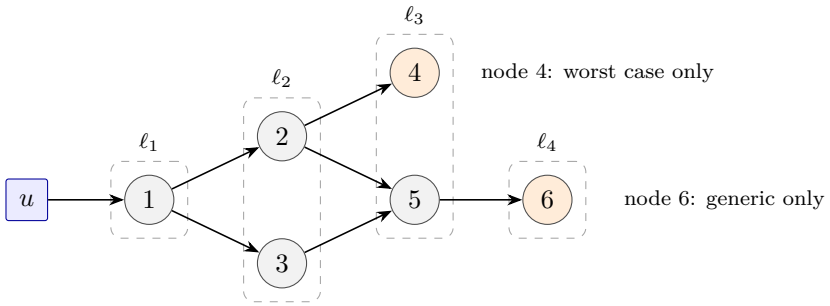


Figure 8.3: The pattern of Remark 8.5: driver 1, state edges $1 \to 2$, $1 \to 3$, $2 \to 4$, $2 \to 5$, $3 \to 5$, $5 \to 6$, and the distance layers $\ell_1 = \{1\}$, $\ell_2 = \{2, 3\}$, $\ell_3 = \{4, 5\}$, $\ell_4 = \{6\}$ of this input-connected HDAG. The two edges into node 5 carry the weight products whose sum $a_{21}a_{52} + a_{31}a_{53}$ is the entry $[\mathcal{C}_F]_{5,3}$, and every other multi-term of the pattern is a multiple of it. The shading marks the two nodes on which the readings of the FSSCS part: node 4 is controllable exactly at the realizations where that sum vanishes (the worst case named in the drawing), and node 6 exactly at the generic ones.

8.3 Fixed Nodes

A subspace answers "which directions", while the node-level question is "which nodes". The translation is the standard basis: node k is individually controllable exactly when its own axis e_k lies in the subspace. Membership of a skew direction like $a_{21}e_2 + a_{31}e_3$ controls a *combination* of nodes, never each node separately.

Definition 8.6 (FSC and FSSC nodes)**.** A state node k is a *fixed structurally controllable* (FSC) node if $e_k \in \mathcal{C}_\Lambda^F$, and a *fixed strongly structurally controllable* (FSSC) node if $e_k \in \mathcal{C}_\Gamma^F$.

The literature also calls these SC and SSC *states* [29, 4]. The name *SSC node* is used differently here (Remark 8.20). Three consequences of the definitions are worth recording at once.

1. *Inclusion.* Every FSSC node is an FSC node, since $\mathcal{C}_\Gamma^F \subseteq \mathcal{C}_\Lambda^F$. The converse fails: diamond node 4 below is FSC and not FSSC.
2. *Counts versus dimensions.* The number of FSSC nodes can be strictly smaller than $\dim \mathcal{C}_\Gamma$: in Example 8.3 the worst case controls a 2-dimensional plane, yet only node 1 is FSSC. Dimensions count directions, while fixed nodes count *axes*.
3. *Drivers are fixed.* A driver node already has an input, and duplicating an input changes nothing, so drivers pass every test below trivially [30].

On the running examples: the rotating plane has FSC nodes $\{1, 2, 3\}$ (SC pattern) but only FSSC node 1. The diamond has FSC nodes $\{1, 4\}$ and FSSC node 1, while the twins $2, 3$ (controllable only in the combination $a_{21}e_2 + a_{31}e_3$) are fixed in neither sense. It remains to *compute* these sets without computing the subspaces, which is the subject of the next two sections.

8.4 Input-Addition Tests

The algebraic characterization compares two patterns: if node k were already controllable in every generic realization (resp. in *every* realization), then giving it its own input should add nothing.

Theorem 8.7 (Fixed-node tests)**.** *Let the pattern graph be input-connected (Definition 4.13) and let* $B_k = [\,B,\ e_k\,]$ *denote the input matrix with a dedicated input added at node* k*. Then* k *is an FSC node if and only if the addition does not increase* $\dim \mathcal{C}_\Lambda$ *[29]. On the SSC side the same test is necessary but not sufficient: if* k *is an FSSC node the addition does not increase* $\dim \mathcal{C}_\Gamma$*, so a strict increase rules* k *out, while a non-increase decides nothing [12].*

Remark 8.8 (No converse on the SSC side). Attach a new node 5 to the diamond by the edge $2 \to 5$ (Fig. 8.4). Its $\mathcal{C}_F$ has third column $(a_{42}a_{21} + a_{43}a_{31})e_4 + a_{52}a_{21}e_5$, whose e_5 entry is a single-term, so $\operatorname{rank} \mathcal{C}_F = 3$ at every realization and an input at 5 leaves $\dim \mathcal{C}_\Gamma$ at 3, yet $e_5 \notin \mathcal{C}_\Gamma^F$, since the realizations with $a_{42}a_{21} + a_{43}a_{31} \neq 0$ do not contain it. The exact criterion is the subspace equality of Lemma B.3: that adjoining e_k to B leaves the controllable subspace unchanged at *every* realization, which a comparison of dimensions cannot see.

Both proofs (deferred to Sections A.4 and B.4) run through the invariance of the controllable subspace under A_F: if e_k already lies in the subspace at every realization that the relevant intersection runs over (the generic ones on the SC side, all of them on the SSC side), then all its propagations $A_F^j e_k$ do too, so the augmented pair spans nothing new. That argument gives "fixed node $\Rightarrow$ no increase" on both sides. Only the SC side has a converse, and it turns on $\dim \mathcal{C}_\Lambda$ being a *maximum*: a single generic realization missing e_k already lifts the augmented maximum by one. Since $\dim \mathcal{C}_\Gamma$ is a *minimum*, a realization missing e_k need not move it, and the implication does not reverse.

Both readings compare dimensions, which is why Chapter 6 had to come first. Each side inherits the computability of its own dimension:

- *FSC side: fully computable.* $\dim \mathcal{C}_\Lambda$ is the maximum disjoint stem-and-cycle cover count (Theorem 6.4), a polynomial matching computation. Running it once for the pattern and once per candidate node settles all FSC nodes in $O(n\,(n + |\mathcal{E}|))$ time [29, 30].
- *FSSC side: a necessary condition only, computable where* $\dim \mathcal{C}_\Gamma$ *is.* The worst-case dimension has no general formula (Section 6.4). On the hierarchical DAGs (Definition 4.21) it is exact via the subgraph reduction of [12], so there the comparison can be carried out, but it only rules nodes out (Remark 8.8), and in general it is open, exactly as open as the dimension itself. Deciding FSSC membership needs Lemma B.3.

Example 8.9 (Testing the diamond)**.** The diamond has $\dim \mathcal{C}_\Lambda = 3$ and $\dim \mathcal{C}_\Gamma = 2$ (Examples 6.2 and 6.5), and it is an HDAG (Definition 4.21, distance layers $\{1\}, \{2, 3\}, \{4\}$), so both dimensions are computable. *Node* 4*, FSC test:* with inputs at 1 and 4, disjoint stems can cover at most $\{1, 2, 4\}$ or $\{1, 3, 4\}$ (the new stem at 4 is the single node 4, since 4 is a sink, and a stem from u can then reach at most two of $\{1, 2, 3\}$), so the cover count stays 3: node 4 *is* FSC. *Node* 4*, FSSC test:* for the augmented pair $B_4 = [\,e_1, e_4\,]$ the columns e_1, e_4, $a_{21}e_2 + a_{31}e_3$ are independent for *every* realization (the cancellation $a_{42}a_{21} = -a_{43}a_{31}$ makes the old third column vanish but not the new input column), so the worst case rises from 2 to 3. A strict increase is the direction Theorem 8.7 does decide: some realization must have been missing e_4, so node 4 is *not* FSSC. *Node* 2 *(and by symmetry* 3*):* an input at 2 lets stems $u \to 1 \to 3$ and $2 \to 4$ cover all four nodes, raising $\dim \mathcal{C}_\Lambda$ to 4: not FSC, hence not FSSC by the inclusion. The SSC side confirms it directly: with $B_2 = [\,e_1, e_2\,]$ the four columns e_1, e_2, $a_{21}e_2 + a_{31}e_3$, $a_{42}e_4$ have determinant $a_{31}a_{42}$, a single monomial, so the worst case rises from 2 to 4. *Node* 1*:* a driver, fixed in both senses. Conclusion: FSC nodes $\{1, 4\}$ and FSSC node $\{1\}$, matching the subspaces of Example 8.4 axis for axis.

8.5 Graph Criteria

The comparisons of Theorem 8.7 still talk about dimensions. The graph criteria read the answer off the picture directly, one per side, the worst-case one deciding not the FSSC nodes but the related class of Definition 8.16 (Remark 8.20).

8.5.1 FSC Nodes on Hierarchical DAGs: Matched in Every Maximum Disjoint-Stem Set

Let the pattern be an HDAG (Definition 4.21) whose drivers are its source nodes. Its *layers* can be built in two ways, and the HDAG hypothesis is what makes the two agree. Recursively: ℓ_1^{rec} collects the state nodes with no incoming state edges (all drivers among them). Remove them and repeat, so that every edge runs from a lower-index layer to a higher one [30]. By distance: ℓ_k collects the nodes at input-distance k. The recursive layering works on any DAG, but there an edge may skip layers and the two then differ. On an HDAG every walk into a node has one length, so no edge skips a layer and $\ell_k^{\text{rec}} = \ell_k$ for every k (the diamond's $\{1\}, \{2,3\}, \{4\}$ is both). Everything below writes the bare ℓ_k and rests on that agreement.

For a layer ℓ_k, a *maximum disjoint-stem set for* ℓ_k is a set of disjoint stems (at most one per driver, since two stems from one driver share it) that covers as many nodes of ℓ_k as possible. The nodes of ℓ_k it covers are its *matched nodes*, a per-layer notion, not the matching of Definition 7.5. Several such sets may be feasible, with different matched sets.

Theorem 8.10 (Matched-node criterion)**.** *Let the pattern be an HDAG (Definition 4.21) whose drivers are its source nodes, so that the recursive layers above are the distance layers. Then node $i \in \ell_k$ is an FSC node if and only if i is matched in* every *feasible maximum disjoint-stem set for ℓ_k [30].*

Corollary 8.11 (Single driver)**.** *On an HDAG with one driver, node $i \in \ell_k$ is an FSC node if and only if it is alone in its layer, $|\ell_k| = 1$.*

Remark 8.12 (Why the hierarchy is assumed). On $1 \to 2$, $1 \to 3$, $2 \to 3$, $2 \to 4$ with driver 1 (Fig. 8.4), edge $1 \to 3$ skips a layer: recursion gives $\ell_1^{\text{rec}} = \{1\}$, $\ell_2^{\text{rec}} = \{2\}$, $\ell_3^{\text{rec}} = \{3,4\}$, the distance layers are $\ell_1 = \{1\}$, $\ell_2 = \{2,3\}$, $\ell_3 = \{4\}$. Node 2 is alone in its recursive layer, so Corollary 8.11 without the hypothesis would call it FSC, yet a driver at 2 frees the old stem to run $1 \to 3$ while $2 \to 4$ takes the rest, lifting $\dim \mathcal{C}_\Lambda$ to 4: node 2 is not FSC, though [30] states the criterion for general layered DAGs. Reading the criterion on the distance layers instead does not repair it: node 4 is alone in ℓ_3, yet a driver there frees the old stem to run $1 \to 2 \to 3$ and lifts $\dim \mathcal{C}_\Lambda$ to 4 as well. Here the FSC node is 1 alone. The hypothesis does not prefer one layering to the other, but makes the two agree.

The intuition is specific to HDAGs. Because the layers are the distance layers, a stem meets ℓ_k in at most one node and does so at its k-th step, so a node matched in every maximum disjoint-stem set for ℓ_k is already covered: an input added there re-traces an existing stem and covers nothing new. A node missed by *some* maximum disjoint-stem set can instead be covered by a new stem from an added input. Both halves need the two layerings to agree: once an edge skips a layer, the added input can send the old stem down a different path and cover more (Remark 8.12). Proofs in Section A.5. Layer by layer the criterion is a disjoint-path computation: connecting a super-source to the drivers and a super-sink to ℓ_k turns it into the r-vertex-disjoint-paths problem, where r is the number of drivers. The problem is solvable in

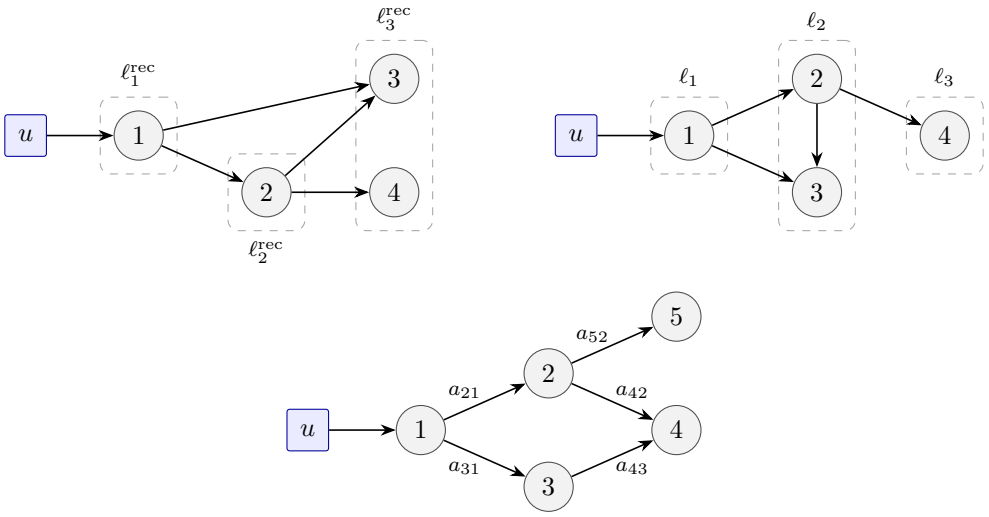


Figure 8.4: Two hypotheses of this chapter, drawn. Top: the pattern of Remark 8.12 ($1 \to 2$, $1 \to 3$, $2 \to 3$, $2 \to 4$, driver 1), the same graph laid out twice, once by the recursive layering ℓ^{rec} and once by the distance layering ℓ. The edge $1 \to 3$ skips a layer and the two readings disagree. Neither of them decides FSC membership correctly, and Theorem 8.10 assumes the hierarchy of Definition 4.21, which makes them agree. Bottom: the diamond with a new node 5 joined by the edge $2 \to 5$ (Remark 8.8), the pattern on which the FSSC half of Theorem 8.7 holds in one direction only.

$O(n + |\mathcal{E}|)$ for one driver and $O(|\mathcal{E}|\, n^{r-1})$ for $r \geq 2$ drivers on DAGs, polynomial for any fixed number of drivers [30].

Example 8.13 (The diamond, by layers)**.** The diamond is an HDAG (Definition 4.21 and Example 4.22) driven at its only source, so the criterion applies: single driver, layers $\{1\}, \{2, 3\}, \{4\}$ by recursion and by distance alike. Corollary 8.11: nodes 1 and 4 are alone in their layers and are therefore FSC, while the twins share ℓ_2 and are not. Via Theorem 8.10 directly: the maximum disjoint-stem sets for ℓ_2 are the single stems $u{\to}1{\to}2$ and $u{\to}1{\to}3$, with matched sets $\{2\}$ and $\{3\}$, so the intersection is empty. For ℓ_3 both $u{\to}1{\to}2{\to}4$ and $u{\to}1{\to}3{\to}4$ match $\{4\}$, so the intersection is $\{4\}$. Same outcome as Example 8.9, no algebra used.

Example 8.14 (Two drivers, two feasible maximum sets)**.** The diamond reaches Theorem 8.10 only through Corollary 8.11. The criterion itself needs two drivers. Take $\mathcal{V}^S = \{1, \ldots, 5\}$ with drivers 1 and 2 and edges $1 \to 3$, $2 \to 4$, $2 \to 5$. The distance layers are $\ell_1 = \{1, 2\}$ and $\ell_2 = \{3, 4, 5\}$, every edge runs from the first to the second, and the drivers are the source nodes: an HDAG (Definition 4.21), so the criterion applies. Two disjoint stems can reach ℓ_2, and exactly two feasible maximum disjoint-stem sets for it exist, namely $\{u_1{\to}1{\to}3,\ u_2{\to}2{\to}4\}$ and $\{u_1{\to}1{\to}3,\ u_2{\to}2{\to}5\}$, with matched sets $\{3, 4\}$ and $\{3, 5\}$. Their intersection is $\{3\}$, so node 3 is an FSC node and nodes 4 and 5 are not. The two drivers are fixed as well, so the FSC nodes are $\{1, 2, 3\}$. The input-addition test (Theorem 8.7) agrees: the best disjoint stem set covers four of the five nodes, so $\dim \mathcal{C}_\Lambda = 4$ (Theorem 6.4). An added input at 3 leaves the count at 4, while one at 4 or at 5 raises it to 5.

Remark 8.15 (Layers decide target control too). The layer count settles more than full-state questions. Suppose only a *target set* of states must be steered, the setting of Section 4.5, with condition $\operatorname{rank}(C\,\mathcal{C}_F) = q$. On a tree rooted at a single driver, every node is reached by a unique walk, so the distance layers are clean (a tree is the simplest hierarchical DAG of

Definition 4.21), and the same one-representative-per-layer counting gives the classical *k-walk criterion*: the target set is generically controllable iff *each layer contains at most one target node* [13]. Two targets in one layer collide in the same column of $C\,\mathcal{C}_F$ for the same reason the twins of Example 8.13 share one direction.

8.5.2 Worst-Case Conditions: Shortest Control Paths

The SSC-side counterpart localizes the dedicated-node conditions of Theorem 7.9 to a single node. Requiring the dedicated node to lie *outside* the subset strengthens what (C1) asks, and where Theorem 7.9 demanded such a node for *every* subset of state nodes, the node-level notion demands one only for the subsets that contain the node in question.

Definition 8.16 (Controllable subset and SSC node)**.** A set $S \subseteq \mathcal{V}^S$ is a *controllable subset* if some node outside S, state or input, sends *exactly one* edge into S, that is, if S has a dedicated node (Definition 7.7) lying outside S. A state node k is an *SSC node* if every $S \subseteq \mathcal{V}^S$ containing k is a controllable subset [23, 31].

On patterns that are trees (Definition 4.19) when edge directions are ignored (*polytrees*), the condition over all subsets collapses to a statement about the paths from the input node to the sinks of Definition 4.20, the state nodes with no out-edges to other state nodes.

Theorem 8.17 (Shortest-path criterion)**.** *Let an input-connected polytree pattern have a single input node u and sinks $t_1, \dots, t_m$. Then a state node $k \in \mathcal{V}^S$ is an SSC node if and only if k belongs to the intersection $\mathcal{V}^u_{\text{int}}$ of the shortest-path graphs (control paths) from u to $t_1, \dots, t_m$ [31].*

An SSC node is thus a node lying on every control path of u. The proof is Section B.5. With several inputs, the union over inputs of these per-input intersections is still a set of SSC nodes (sufficiency) [31]. Computing all the shortest-path graphs costs $O(pmn^2)$, where p is the number of inputs (the m of Eq. (2.1), re-lettered because Theorem 8.17 has bound m to the sinks) and m the number of sinks. The cost is therefore at most $O(n^4)$. And the property transfers down the grid:

Proposition 8.18 (SSC nodes are FSC nodes)**.** *Let an input-connected polytree pattern have a single input node. Then every SSC node is an FSC node. With several inputs the same conclusion holds for every node of the union $\bigcup_j \mathcal{V}^{u_j}_{\text{int}}$ of the per-input intersections [31].*

With one input a polytree is an HDAG, so Theorem 8.10 applies: a node on every control path is matched in every maximum disjoint-stem set for its layer, and an added input cannot raise the cover count, the FSC half of Theorem 8.7 restated on the graph (that proof, and the several-input case, in Section B.6).

Example 8.19 (Pruned diamond, and a gap between the two notions)**.** Delete the diamond's edge $3 \to 4$: $u \to 1$, $1 \to 2$, $1 \to 3$, $2 \to 4$, a polytree with sinks 3 and 4. The control paths are $u{\to}1{\to}3$ (state nodes $\{1,3\}$) and $u{\to}1{\to}2{\to}4$ (state nodes $\{1,2,4\}$). Their intersection is $\{1\}$, so node 1 is the only SSC node (Theorem 8.17), and by Proposition 8.18 it is FSC. The other node classes, for contrast: layers are again $\{1\},\{2,3\},\{4\}$, so the FSC nodes are $\{1,4\}$. Since $\mathcal{C}_F = [\,e_1,\ a_{21}e_2 + a_{31}e_3,\ a_{42}a_{21}e_4,\ 0\,]$ has a single-term entry in its third column, every realization has rank exactly 3 and $\mathcal{C}^F_\Gamma = \operatorname{span}\{e_1, e_4\}$: node 4 is FSSC as well. Two lessons. First, the two worst-case node notions do not coincide: node 4 is FSSC, yet $S = \{2,3,4\}$ has node 1 sending two edges into it and no outside dedicated node, so 4 is not an SSC

node. The subset notion of Definition 8.16 asks that *every* subset of state nodes containing 4 be a controllable subset. Membership of the single axis e_4 in the FSSCS asks something else, and neither reading is claimed here to imply the other (Remark 8.20). Second, the polytree hypothesis matters: the diamond itself is *not* a polytree (undirected cycle 1 2 4 3), so Theorem 8.17 does not apply to it, but the subset definition still does, and a direct check ($\{2,3\}$ and $\{2,3,4\}$ have no outside dedicated node, while every subset containing 1 is served by u) gives SSC node $\{1\}$, agreeing with the FSSC computation of Example 8.9.

Remark 8.20 (A terminology caution). The naming here is not universal. This book reserves *SSC node* for the subset/path notion of Definition 8.16 [23, 31] (every subset containing k controllable) and *FSSC node* for the membership $e_k \in \mathcal{C}_\Gamma^F$ (Definition 8.6), which the min-dimension comparison of Theorem 8.7 can rule out but not confirm. The cited survey [4] attaches the name "SSC node" to *that* min-dimension property, so the survey's "SSC node" is this book's *FSSC node*. The two are genuinely different tests: pruned-diamond node 4 is FSSC but not an SSC node (Example 8.19). No implication in the other direction is asserted in this book. The published results relate the subset/path notion to the *FSC* nodes (Proposition 8.18) and leave its relation to the FSSC nodes open [31].

8.6 Node Ranking and Grid Completion

The node classes assemble into a ranking of the nodes by robustness to weight uncertainty, that is, by how reliably each can be controlled when only the pattern is known:

1. **FSSC nodes** (their own axis survives the intersection over every realization) and **SSC nodes** of Definition 8.16, for which every subset of state nodes containing them is a controllable subset (on a polytree, lying on every control path). Two different tests, not two names for one (Remark 8.20).
2. **FSC nodes**, controllable in every generic realization, but lost at some realization.
3. **reachable non-fixed nodes**, contributing to the dimension only jointly (the diamond's twins: one direction, two nodes).
4. **unreachable nodes**, never controllable.

The first level sits above the second: an FSSC node is an FSC node, since $\mathcal{C}_\Gamma^F \subseteq \mathcal{C}_\Lambda^F$, and an SSC node is one under the hypotheses of Proposition 8.18. The two first-level tests remain distinct from each other (Remark 8.20). On the diamond, whose FSSC and SSC classes are both $\{1\}$ while its FSC class is $\{1,4\}$, the ranking reads $1 \succ 4 \succ \{2,3\}$. The ranking has a direct practical use: assign the tasks that *must* be executed to the FSSC nodes, which keep their own axis at *every* realization, and (under the hypotheses of Proposition 8.18) to the SSC nodes, which instead meet the subset condition of Definition 8.16, the dedicated-node conditions of Theorem 7.9 localized to a single node. An FSC node, by contrast, tolerates parameter variation only among the generic realizations and is lost at some realization. The remaining nodes are candidates for redesign [31, 4]. And Theorem 8.7 gives the ranking a second reading that opens Part III: a *new input that increases a dimension* always sits on a node that is not fixed, exactly the non-FSC nodes on the SC side, while on the SSC side a strict increase of $\dim \mathcal{C}_\Gamma$ rules the node out as FSSC and a non-increase decides nothing (Remark 8.8). The non-fixed nodes are the effective input sites, so the next question is unavoidable: how many inputs does a network need, and where should they go (Chapter 9)?

With the node row the book's summary grid is complete. Its rows are the four analysis levels rather than the algebra/graph pair of the per-chapter tables: the first two rows state

their levels algebraically (the determinant row for a single input, Proposition 5.4), the third on the graph, and the node row carries both forms. The node row is where the two sides diverge: on the SC side both cells are exact (the graph cell under the hierarchy hypothesis of Definition 4.21), while on the SSC side the min-dimension comparison is only necessary (Remark 8.8) and the shortest-path condition, exact on a polytree with one input, decides the related but distinct class of *SSC nodes* (Remark 8.20):

	SC (almost all)	**SSC** (all)
determinant	$\det \mathcal{C}_F \not\equiv 0$	$\det \mathcal{C}_F$ never zero
subspace	$\dim \mathcal{C}_\Lambda = n$	$\dim \mathcal{C}_\Gamma = n$
whole graph	input-connected + stems+cycles cover (Theorem 7.2)	dedicated subsets (Theorem 7.9); ZFS under free self-loops (Theorem 7.14)
node	FSC: max-dim test (Theorem 8.7); on an HDAG, matched in every maximum disjoint-stem set (Theorem 8.10)	FSSC: min-dim test, necessary only (Theorem 8.7); SSC nodes: shortest-path condition (Theorem 8.17)

Sources. The fixed subspace and the FSC input-addition test originate with [29], with a separator-based follow-up in [32], and $\dim \mathcal{C}_\Lambda$ computability is [19]. The matched-node criterion and its algorithms are the author's [30], stated there for layered DAGs and restricted here to hierarchical ones, a hypothesis neither that paper nor the dissertation [1] carries, and one the general form cannot do without (Remark 8.12). The FSSCS, the min-dimension comparison, and the HDAG computation of $\dim \mathcal{C}_\Gamma$ are [12]. The SSC-node notion (dedicated nodes and controllable subsets) originates in [31] and is restated in [23], and the shortest-path criterion and the ranking application are also [31]. The survey [4] organizes the node-level theory, and the orientation figures and the unified treatment follow the dissertation [1].

Exercises

Exercise 8.1 (The two intersections)**.** State both intersections of Definition 8.2 and explain, in two sentences, why $\mathcal{C}_\Gamma^F \subseteq \mathcal{C}_\Lambda^F$ needs no hypothesis.

Exercise 8.2 (The FSSC half)**.** Carry out the FSSC half of Theorem 8.7 at node 4, first for the diamond and then for the pruned diamond of Example 8.19. Give $\dim \mathcal{C}_\Gamma$ before and after the addition in each case, and say what each outcome decides. [Hint: one of the two decides nothing. Compare Remark 8.8.]

Exercise 8.3 (Matched nodes, two drivers)**.** Add the edge $1 \to 4$ to the pattern of Example 8.14, keeping the drivers 1 and 2. Check that the result is still an HDAG (Definition 4.21) whose drivers are its source nodes, list the feasible maximum disjoint-stem sets for ℓ_2, and give the FSC nodes by Theorem 8.10.

Exercise 8.4 (When the hierarchy fails)**.** Give node 3 of Example 8.14 its own input, so that the drivers are 1, 2 and 3. Show that the augmented pattern is no longer an HDAG, so that Theorem 8.10 does not apply to it. Which test settles the FSC status of node 3 in the original pattern without any hierarchy hypothesis, and what does it give?

Exercise 8.5 (Two realizations pin the FSSCS)**.** The intersection of Definition 8.2 runs over the whole family, but two members of it can already settle the answer. Exhibit two realizations

of the pattern of Remark 8.5 whose controllable subspaces meet in $\operatorname{span}\{e_1\}$ alone, and say why e_1 survives at every realization, so that $\mathcal{C}_\Gamma^F = \operatorname{span}\{e_1\}$ follows without enumerating the family. Check your answer against Section B.7. [Hint: take one realization with $a_{21}a_{52} + a_{31}a_{53} \neq 0$ and one on which the weights out of node 1 are tuned against those into node 5 so that the sum vanishes.]

Exercise 8.6 (FSSC nodes against SSC nodes)**.** Show that every node of the four-node network (Fig. 4.1) is an FSSC node, yet only nodes 1 and 2 are SSC nodes in the sense of Definition 8.16. [Hint: for the second half one two-element subset does all the work.]

Exercise 8.7 (All four node classes)**.** For the pattern $u\to1$, $1\to2$, $2\to3$, $2\to4$: find the SSC nodes by Theorem 8.17, the FSC nodes by Corollary 8.11, and check that Proposition 8.18 holds. Then compute $\mathcal{C}_F$, $\mathcal{C}_\Lambda^F$ and $\mathcal{C}_\Gamma^F$, and compare all four node classes.

Exercise 8.8 (Ranking the pruned diamond)**.** State the four levels of the ranking of Section 8.6 and place each node of the pruned diamond (Example 8.19) in one of them. Which levels come out empty?

Part III

Input Placement: Synthesis

Chapter 9

Minimum Input Problem

Part III reverses the direction of every question so far: instead of judging a given input matrix, we choose one. The first design question is cardinality (how few inputs, and attached where, make the network structurally or strongly structurally controllable), and its two answers bracket the field. For SC the answer is a maximum matching: polynomial and exact, with the minimum count equal to the number of unmatched nodes, or one when the matching is perfect. For SSC the answer is a minimum zero forcing set, whose size is NP-hard to compute in general, the cost of a worst-case guarantee. Solvability in polynomial time returns only for structured classes. The same limitation applies on both sides: the minimum fixes how many *inputs but barely* where*. Many placements achieve that count, and Chapter 10 must choose among them.*

9.1 From Analysis to Synthesis

Every criterion of Part II took the pair (A_F, B) as given and returned an answer. Design inverts the arrow. The pattern A (the physics of the network, its wires and couplings) is fixed by the application and not ours to change. What we place is B, the set of nodes we agree to actuate. Reread in this light, each theorem of Chapters 7 and 8 becomes a *feasibility constraint* on that choice. Lin's cover condition (Theorem 7.2) says a driver set is SC-feasible iff, together with the pattern, it leaves no dilation and no unreachable node. Mayeda's pair (Theorem 7.9) says it is SSC-feasible iff every subset has a dedicated node (C1), one lying outside the subset in the cycle-like case (C2). A *driver set*, written $\mathcal{D}$, is a set of state nodes $\mathcal{D} \subseteq \mathcal{V}^S$, each carrying its own dedicated input column e_k, and the input matrix is then $B = [\, e_k : k \in \mathcal{D} \,]$. The minimum input problem asks for a smallest feasible driver set:

Problem 9.1 (Minimum input problem, MIP)**.** Given a pattern graph, find a driver set $\mathcal{D}$ of minimum cardinality such that the family is SC (respectively SSC), and locate its nodes.

Two questions hide inside one: *how many* inputs, and *where*. The two controllability notions answer them on opposite ends of the tractability scale, and the rest of the chapter takes each in turn.

9.2 Minimum Inputs for Structural Controllability

On the SC side the matching vocabulary of Section 7.2 already did the combinatorial work. Recall (Definition 7.5) that a matching on the state subgraph is a set of edges sharing no tail and no head, and that a state node is *matched* if it is some matched edge's head. Following matched edges tail-to-head decomposes the matched nodes into disjoint paths and cycles. The unmatched nodes are exactly the nodes at which those paths *start*: the nodes with no matched in-edge, at which a stem must be started from outside. Placing one input at each of them turns the matching into a disjoint set of stems and cycles covering all state nodes, which is the cover half of condition (ii) of Theorem 7.2. Counting the unmatched nodes is therefore counting the inputs, and the count is a matching invariant. The other half of that condition, input-connectedness, does not come with it (Remark 9.2).

Theorem 9.1 (Minimum inputs for SC)**.** *For a pattern graph on state nodes $\mathcal{V}^S$, let ν be the size of a maximum matching on the state subgraph. Then the minimum number of inputs rendering the family SC equals the number of unmatched state nodes, $|\mathcal{V}^S| - \nu$, or one if that number is zero (a perfect matching) [3].*

Remark 9.2 (Where the inputs attach). Theorem 9.1 counts inputs, and one input may be linked to several nodes. Attaching one input to each unmatched node is the standard choice, and since it already supplies the cover, Theorem 7.2 makes it succeed exactly when the result is input-connected. However, a cycle of the matching decomposition has every one of its nodes matched, so it receives no input and can be left unreachable. On $1 \to 2$, $3 \to 4$, $4 \to 3$ (Fig. 9.1) the matching $\{1{\to}2,\ 3{\to}4,\ 4{\to}3\}$ is maximum, so $\nu = 3$ and node 1 is the only unmatched node. An input there gives $\mathcal{C}_F = [\,e_1,\ a_{21}e_2,\ 0,\ 0\,]$, of rank 2. The repair costs no input: link the *same* input to node 3 as well, so that $B = [\,e_1 + e_3\,]$ and $\det \mathcal{C}_F = a_{21}a_{34}^2a_{43}^3$, a single non-zero monomial. So the count of Theorem 9.1 is a count of inputs. A minimum *driver set* in the sense of Problem 9.1, one dedicated column per node, can be strictly larger.

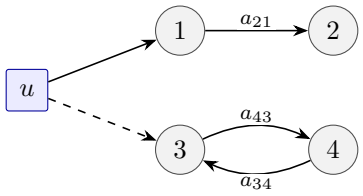


Figure 9.1: The pattern of Remark 9.2: the state edge $1 \to 2$ and the cycle $3 \to 4 \to 3$, which no edge joins. The matching $\{1{\to}2,\ 3{\to}4,\ 4{\to}3\}$ is maximum, so $\nu = 3$ and node 1 is the only unmatched node. Theorem 9.1 therefore returns one input. Attached at node 1 alone (solid) it leaves the cycle unreachable and $\operatorname{rank} \mathcal{C}_F = 2$. Linking the *same* input to node 3 as well (dashed) costs no second input and gives $\det \mathcal{C}_F = a_{21}a_{34}^2a_{43}^3$, a single non-zero monomial.

The count is independent of *which* maximum matching is used (all have the same size), so the *number* is canonical even though the *location* is not, a non-uniqueness Section 9.4 returns to. Because a maximum matching of the bipartite graph representation is computable in $O(\sqrt{n}\,|\mathcal{E}|)$ time, the whole minimum problem for SC is polynomial: the tractable end of the field. The theorem is due to Liu, Slotine, and Barabási [3], who read it as the "minimum driver nodes" that render a complex network controllable. It is the design-side form of Lin's analysis theorem, connected through the matching–cover correspondence of Section 7.2.

Example 9.3 (The four-node network already attains the minimum)**.** On the state subgraph of Fig. 4.1 the edge set $\{1{\to}2,\ 2{\to}3,\ 3{\to}4\}$ is a matching of size three (Example 7.6), and no matching of size four exists, so $\nu = 3$ and $|\mathcal{V}^S| - \nu = 1$. The one unmatched node is node 1: its only in-edge is the cycle edge $3{\to}1$, so leaving it unmatched is the natural choice. Since $\nu = 3$ and $|\mathcal{V}^S| = 4$, every maximum matching leaves *exactly one* node unmatched, but the exposed node need not be a usable site. The alternative matching $\{1{\to}2,\ 2{\to}3,\ 3{\to}1\}$ of Example 7.6 exposes node 4, which has no out-edges, so a single input there gives $\mathcal{C}_F = [\,e_4,\ 0,\ 0,\ 0\,]$, of rank 1: the matched nodes $1, 2, 3$ form a cycle of the decomposition and stay unreachable (Remark 9.2). The third maximum matching, $\{1{\to}2,\ 2{\to}4,\ 3{\to}1\}$, exposes node 3, and that site does work: an input at 3 gives $\det \mathcal{C}_F = a_{13}^3 a_{21}^2 a_{42}$, again a single monomial. Theorem 9.1 thus places a single input, here at node 1. This is precisely the driver node of the running network, whose one input we took for granted throughout Part II. The minimum problem now *explains* that choice rather than assuming it. And the input gives the strongest possible outcome: the stem $u{\to}1{\to}2{\to}3{\to}4$ covers every node, so the family is SC, indeed SSC, with the single-monomial determinant $a_{21}^3 a_{32}^2 a_{43}$ of Example 4.25. One input, optimally placed, and nothing to improve.

Example 9.4 (The diamond needs two)**.** The diamond (Fig. 6.1) behaves in the opposite way. Its state subgraph has edges $1{\to}2$, $1{\to}3$, $2{\to}4$, $3{\to}4$, and every matching must choose between the heads 2 and 3, which share their only tail 1: a maximum matching such as $\{1{\to}2,\ 3{\to}4\}$ has size two and leaves nodes 1 and 3 unmatched (Example 7.6). Hence $\nu = 2$, $|\mathcal{V}^S| - \nu = 2$, and the minimum is *two* inputs. This is the matching form of the dilation $S = \{2, 3\}$, $N^{\text{in}}(S) = \{1\}$, that already denied the diamond SC with one input (Example 7.3). Node 1, a source with no in-edges, must be driven directly. The second input goes to whichever twin the chosen matching left unmatched. The two parallel paths to node 4 demand two independent inputs: no single stem can cover both twins, so no single input can. We verify below (Section 9.4) that either $\{1, 2\}$ or $\{1, 3\}$ does the job, and neither is forced.

9.3 Minimum Inputs for Strong Structural Controllability

Strengthening SC to SSC replaces the matching by its worst-case counterpart from Section 7.4. There, forcing was a decision procedure: a driver set is SSC-feasible (under the free-self-loop hypothesis of Theorem 7.14) iff it is a zero forcing set, one whose color-change process blackens all of $\mathcal{V}^S$. Minimizing the number of inputs is therefore minimizing the size of a zero forcing set.

Definition 9.5 (Zero forcing number)**.** The *zero forcing number* $Z(\mathcal{G})$ of a pattern graph is the minimum cardinality of a zero forcing set (Definition 7.13).

Theorem 9.6 (Minimum inputs for SSC)**.** *Under the standing free-self-loop hypothesis of Theorem 7.14, the minimum number of driver nodes rendering the family SSC equals the zero forcing number $Z(\mathcal{G})$, and a minimum driver set is any minimum zero forcing set [26].*

The identification of the minimum driver count with $Z(\mathcal{G})$ is due to Trefois and Delvenne [26], who also cast a zero forcing set as a *constrained matching*, the uniqueness-of-matching condition that Example 7.16 met on the chain. Uniqueness is exactly what worst-case robustness costs: an SC minimum tolerates many maximum matchings, whereas an SSC minimum demands a matching that is the only one on its own node set.

That extra demand is expensive. Deciding whether a driver set forces is polynomial (apply the color-change rule until no more white nodes can be colored black), but searching for the *smallest* forcing set is a combinatorial optimization over 2^n candidate sets, and no polynomial algorithm is known, nor expected.

Remark 9.7 (NP-hardness of the SSC minimum). Computing the zero forcing number of a general graph is NP-hard, and so therefore is the SSC minimum input problem in general [27, 26]. This is not a gap in the present treatment but a feature of the worst-case guarantee: SSC requires controllability for *every* realization, and finding the lowest-cost way to obtain that guarantee is as hard as the underlying set-cover-like problem. The same hardness applies to exact (non-structural) minimal controllability [33]. The practical consequences are the usual ones (exact solution only for small or structured instances, approximation or heuristics otherwise), and they emphasize the contrast with the polynomial SC side of Theorem 9.1.

Tractability returns on structured classes, where the graph's structure determines the forcing process. The following are stated at the level of results. The constructions and proofs are the author's [23] and the survey's [4].

Remark 9.8 (Polynomial special classes). For several families the minimum forcing set (hence the SSC minimum input set) is computable in polynomial time, by decomposing the network into basic controllable components and *merging* them [23]. These component results are stated for the undirected diffusively-coupled networks of Remark 9.9 (every state edge two-way, every node self-looped), so the "end node" and "two inputs" readings below are the undirected-with-self-loops statements. In contrast, a *directed* cycle is a single forcing chain and needs only one input. A *bridge edge* below is an edge joining two otherwise disjoint components of the decomposition. The edge that attaches a cycle bud to its stem in Lin's cacti (Theorem 7.2) is one instance. The decompositions below are the special case in which no two components are joined by more than one such edge:

- *Paths.* A state subgraph that is a path is SSC iff a single input is attached to an *end* node (Definition 4.5). The minimum is one. The path is the simplest case: its forcing chain is the stem of Example 7.16.
- *Trees.* A state subgraph that is a tree is SSC iff it decomposes into SSC path components joined by single bridge edges. The minimum input count equals the number of such paths, a path-cover count computable in polynomial time [34, 23].
- *Cycles.* A cycle needs *two* suitably placed inputs (adjacent driven neighbors), never one: a lone input on a cycle leaves a subset with no dedicated node.
- *Pactus.* A *pactus* (a generalization of the cactus in which paths and cycles are glued along bridge edges, with several of them allowed between two components) is SSC if each component is, under the single-bridge-edge restriction stated above. A merging procedure assembles the minimum external inputs by reusing internal *component input nodes*: state nodes that, seen from an adjacent component, act exactly like an external input. The merging itself is $O(|\mathcal{V}^S|)$, and only the initial path-cover decomposition inherits the general NP-hardness [23].

Remark 9.9 (Diffusively-coupled networks and the self-loop hypothesis). The free-self-loop hypothesis of Theorems 7.14 and 9.6 is not an idealization but the native regime of *diffusively-coupled* networks (consensus, synchronization, opinion dynamics), whose system matrix is a (modified) Laplacian $\tilde{\mathcal{L}} = \mathcal{L} - \mathcal{S}$, where $\mathcal{S}$ is a diagonal matrix carrying one tunable self-loop weight per agent. There the diagonal is a free parameter, so the zero-forcing count of

Theorem 9.6 is the *exact* minimum, and the merging rules of Remark 9.8 apply verbatim [23]. One caveat, promised long ago: a Laplacian family has *dependent* entries, each diagonal being minus its row sum (Remark 2.11), so the self-loop weights are not wholly free of the off-diagonal ones. The merging analysis is carried out for exactly this modified-Laplacian model, which is why its self-loop freedom is legitimate. For a structurally loop-free pattern the case is the one shown in Example 7.15: forcing is only *sufficient*, so $Z(\mathcal{G})$ is an upper bound on the SSC minimum, and the exact count needs the two-rule / constrained-matching machinery of Remark 7.17. The four-node network is precisely such an instance: it is SSC with one input (Example 9.3), yet the color-change process starting from $\{1\}$ stops with derived set $\{1, 2\}$, so its zero forcing number overcounts.

9.4 Non-Uniqueness

The minimum fixes a number, not a placement. Different maximum matchings expose different unmatched nodes, each a candidate minimum driver set, and by Theorem 7.2 a candidate succeeds exactly when the resulting pattern is also input-connected (Remark 9.2). The same freedom holds for minimum zero forcing sets. The count is canonical. The location is a choice, and Chapter 10 shows that it is not a cost-neutral one.

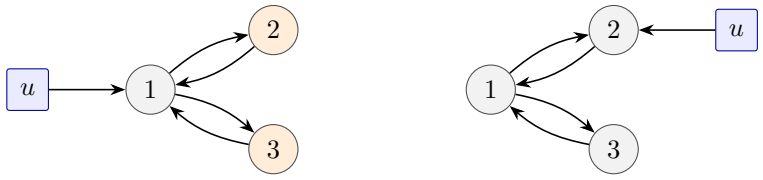
Example 9.10 (Two minima for the diamond)**.** Example 9.4 fixed the diamond's minimum at two inputs, one of them forced at the source node 1. Where does the second go? Two maximum matchings give the two answers: $\{1{\to}2,\ 3{\to}4\}$ leaves $\{1, 3\}$ unmatched, and $\{1{\to}3,\ 2{\to}4\}$ leaves $\{1, 2\}$. Both $\{1, 3\}$ and $\{1, 2\}$ are minimum driver sets, as a direct computation confirms. With inputs at $\{1, 2\}$, $B = [\, e_1,\, e_2\,]$ and the four columns $e_1,\ e_2,\ A_F e_1 = a_{21}e_2 + a_{31}e_3,\ A_F e_2 = a_{42}e_4$ have determinant $a_{31}a_{42}$. With inputs at $\{1, 3\}$ the analogous determinant is $-a_{21}a_{43}$. Each is a single non-zero monomial, so *both* placements are not merely SC but SSC, at every realization. What is *not* free is the source: $\{2, 3\}$ actuates neither node 1 (unreachable from the twins) nor the dilation it sits above, and fails. $\{1, 4\}$ is input-connected yet still fails: with node 4 driven, its own stem is the single node 4 and the u-stem reaches at most two of $\{1, 2, 3\}$, so disjoint stems cover only three nodes (Example 8.9). The sets $\{2, 4\}$ and $\{3, 4\}$ leave node 1 unreachable like $\{2, 3\}$. So the diamond admits exactly two minimum SC driver sets, $\{1, 2\}$ and $\{1, 3\}$: the same count, mirror-image locations, the twin symmetry resurfacing one last time. The minimum constrains how many. Symmetry, not the minimum, narrows the where.

Example 9.11 (The star: placement changes the count)**.** The twin star (Examples 5.5 and 3.9) shows this most sharply (Fig. 9.2). Driven at its *center* (the symmetric, obvious choice), it is uncontrollable: the end nodes $\{2, 3\}$ form a dilation fed through one node, and $\det \mathcal{C}_F \equiv 0$ (Example 5.5). A first repair keeps the center input and adds a second at an end node. With $B = [\, e_1,\, e_2\,]$ the matrix $\mathcal{C}$ has rank three, so *two* inputs restore controllability. It is tempting to stop there and declare that the star needs two inputs. It does not. A maximum matching of the star has size two and leaves exactly *one* node unmatched, so Theorem 9.1 promises a minimum of one, provided it is placed correctly. Drive a single *end node* instead of the center: with $B = e_2$,

$$\det \mathcal{C}_F \;=\; -\,a_{12}^2\, a_{31},$$

a single monomial, non-zero at every realization. One end-node input makes the star not merely SC but SSC. The central placement is what turns $\{2, 3\}$ into a dilation and needs a second input. The matching count gives the correct answer. Here the "where" does not merely

tune the cost: it changes the minimum count from an apparent two to an actual one. It shows as plainly as possible that a design chapter cannot read cardinality off the network without also solving for location.



(a) one input at the center (b) one input at an end node

Figure 9.2: The twin star of Example 9.11 under the two placements of a single input. (a) At the center the end nodes $\{2,3\}$ (shaded) are a dilation with $N^{\text{in}}(\{2,3\}) = \{1\}$ (Definition 7.1) and $\det \mathcal{C}_F \equiv 0$: no realization is controllable, and only a second input, at an end node, restores rank 3. (b) At an end node one input already gives $\det \mathcal{C}_F = -a_{12}^2 a_{31}$, a single non-zero monomial, so the family is SSC. A maximum matching of the star has size two and leaves one node unmatched, so the minimum count of Theorem 9.1 is one, and only the second placement attains it.

So the minimum input problem returns a number and a family of placements that achieve it. The well-matched four-node network takes one input, at node 1 or node 3. The diamond takes two. The star takes one, but only at an end node. Cardinality is settled. Among the placements of minimum size, and the deliberately larger ones, none is yet singled out.

9.5 Minimum Inputs, Two Ways

The chapter's two answers fit the book's running two-column format, now read as a design summary rather than an analysis grid:

	SC (almost all)	**SSC** (all)
minimum count	*inputs*: unmatched nodes, $\lvert\mathcal{V}^S\rvert - \nu$ (one if zero)	*driver nodes*: zero forcing number $Z(\mathcal{G})$
attained by	maximum matching (Theorem 9.1)	minimum ZFS (Theorem 9.6)
cost	polynomial [3]	NP-hard; polynomial on special classes (Remark 9.8)

The two columns count different objects: Theorem 9.1 counts *inputs*, one of which may attach to several nodes, while Theorem 9.6 counts *driver nodes*, one dedicated column each. A minimum driver set on the SC side can therefore be strictly larger than the input count (Remark 9.2). Both columns end at the same point: a count fixed, a placement free. Choosing among the free placements (by control energy, by robustness, by cost) is the optimal placement problem of Chapter 10, where the decision returns to the algebra of Gramians.

Sources. The matching solution of the SC minimum is Liu, Slotine, and Barabási [3], resting on Lin [16]. The survey [4] sets it in the wider MIP taxonomy. The zero-forcing-number identification of the SSC minimum and its constrained-matching form are Trefois and Delvenne [26], on the criterion of [25]. NP-hardness of the SSC minimum is due to [27, 26]. The

polynomial special classes (paths, trees, cycles, and pactus, together with diffusively-coupled networks) are the author's merging-rules paper [23], with the tree/path-cover case going back to [34] and the general (non-structural) minimal-controllability hardness to [33].

Exercises

Exercise 9.1 (Count against placement)**.** State Theorem 9.1 and explain why the count it returns is the same for every maximum matching while the placement it suggests is not.

Exercise 9.2 (Unmatched nodes on three patterns)**.** For each state subgraph below compute ν, the minimum input count of Theorem 9.1, and every set of unmatched nodes a maximum matching can produce: (a) the path $1\to2\to3\to4$, (b) the cycle $1\to2\to3\to1$, and (c) the tree with edges $1\to2$, $1\to3$, $2\to4$.

Exercise 9.3 (A repair that costs no input)**.** Explain, using Remark 9.2, why attaching one input to each unmatched node can still fail, and describe the repair that costs no extra input.

Exercise 9.4 (Exposing the wrong node)**.** The maximum matching $\{1\to2,\ 2\to3,\ 3\to1\}$ of the four-node network exposes node 4. Show that a single input there gives $\operatorname{rank}\mathcal{C}_F = 1$, and name the structural feature responsible. Then verify that linking the *same* input to node 2 as well ($B = [\,e_2 + e_4\,]$) makes the family SSC, by computing $\det\mathcal{C}_F$.

Exercise 9.5 (Five zero forcing numbers)**.** Compute the zero forcing number $Z(\mathcal{G})$ of Definition 9.5 for: the undirected path on four nodes, the undirected cycle on four nodes, the twin star (Example 3.9), the undirected star with three end nodes, and the directed cycle on four nodes. Which of the five need the most inputs, and why does the direction of the cycle's edges change the answer?

Exercise 9.6 (The diamond's forcing sets)**.** Under the free-self-loop hypothesis of Theorem 7.14, find $Z(\mathcal{G})$ and *all* minimum zero forcing sets of the diamond, and compare them with the two minimum SC driver sets of Example 9.10.

Exercise 9.7 (When forcing overcounts)**.** Remark 9.9 states that the four-node network's zero forcing number overcounts. Compute $Z(\mathcal{G})$ for it, list the minimum zero forcing sets, and say by how much the count exceeds the SSC minimum of Example 9.3. Which hypothesis is responsible?

Exercise 9.8 (Merging path components)**.** Take the undirected tree with edges 1–2, 2–3, 3–4, 3–5 and a self-loop of free weight at every node. Decompose it into SSC path components joined by a single bridge edge, as in Remark 9.8, give the resulting minimum input count, and confirm it by exhibiting a minimum zero forcing set.

Chapter 10

Optimal Input Placement and Outlook

Chapter 9 closed on a deliberate limitation: the minimum input count constrains how many *drivers a network needs, but barely* where *they go, and placements of equal size can differ wildly in control effort. This closing chapter supplies the missing metric. The controllability Gramian of Chapter 3 measures a placement by three numbers (average ease, worst-direction energy, reachable volume), each an exact statement about the minimum-energy formula, and a computed comparison on the diamond shows the metrics disagreeing in an instructive way. Optimal placement is then an optimization over driver sets: combinatorial in general, greedy in practice, with guarantees worth knowing at statement level. A short duality section transfers everything to output placement, and four open problems, each scoped as a course project, leave the reader at the open questions of the field.*

10.1 Comparing Placements

Recall the promise made in Chapter 3: the Gramian

$$W(T) = \int_0^T e^{A\tau} B B^\top e^{A^\top \tau}\, d\tau$$

refines *whether* a direction is reachable into *how hard*, through the minimum-energy formula of Theorem 3.4: steering the state from the origin to x_f in time T costs, at best,

$$E(x_f) \;=\; x_f^\top W(T)^{-1} x_f. \tag{10.1}$$

Every question about the *quality* of an input placement is a question about Eq. (10.1), hence about the spectrum of $W(T)$. Three functionals of that spectrum dominate the literature, and each has an exact reading (not an analogy) in terms of Eq. (10.1):

- *Average ease:* for a target drawn uniformly from the unit sphere, the expected energy is $\operatorname{tr}(W^{-1})/n$, and minimizing $\operatorname{tr}(W^{-1})$ minimizes average energy. Its low-cost and popular proxy reverses the inverse: $\operatorname{tr}(W)$ is n times the spherical average of $x_f^\top W x_f$, equivalently the total impulse-response energy $\int_0^T \|e^{A\tau}B\|_F^2\, d\tau$ (a measure of how strongly the inputs excite the state on average). The proxy fails in one important way, exposed below.

- *Worst direction:* over unit targets, the most expensive one costs

$$\max_{\|x_f\|=1} E(x_f) \;=\; \frac{1}{\lambda_{\min}(W)},$$

  so maximizing $\lambda_{\min}(W)$ minimizes worst-case energy. Unlike the trace, this metric *guarantees* controllability: $\lambda_{\min}(W) > 0$ iff (A, B) is controllable, and its value is a usable positive margin rather than a bare yes or no.
- *Reachable volume:* the set of states reachable with unit energy, $\{x : x^\top W^{-1} x \le 1\}$, is an ellipsoid with semi-axes $\sqrt{\lambda_i(W)}$ along the eigenvectors of W. Its volume is proportional to $\sqrt{\det W}$. Maximizing $\log\det W$ maximizes the log-volume reachable at unit energy, spreading effort over *all* directions rather than the worst or the average one. Like $\lambda_{\min}(W)$ and unlike the trace it detects failure: $\log\det W$ is $-\infty$ exactly when W is singular.

Remark 10.1 (Which Gramian? A horizon caveat). For Hurwitz A one may take $T = \infty$ and solve the Lyapunov equation of Chapter 3. The pattern used below, the diamond of Fig. 6.1, is a DAG: its A is *nilpotent*, every eigenvalue is zero, no infinite-horizon Gramian exists ($W(T)$ grows polynomially in T). Two standard repairs: fix a finite horizon T (our choice below, with $T = 1$, for which the integral is exact, since $e^{A\tau}$ is a polynomial in τ) or shift $A \mapsto A - \alpha I$ with α large enough to make the matrix Hurwitz, which discounts late response and restores the Lyapunov argument (the brain-network literature stabilizes similarly, by normalizing the adjacency matrix [35]). Either way the metric *values* depend on the horizon or the shift, but the zero/non-zero outcomes do not, because $\operatorname{Im} W(T) = \operatorname{Im} \mathcal{C}$ for every $T > 0$ (Theorem 3.4).

Example 10.2 (Two placements on the diamond, compared)**.** Take the diamond of Fig. 6.1 with all weights 1,

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix},$$

and compare two placements of the same size two (Fig. 10.1): drivers $\mathcal{D} = \{1, 2\}$, that is $B = [\,e_1 \;\; e_2\,]$, against $\mathcal{D} = \{1, 4\}$. Since $A^3 = 0$, the matrix exponential terminates, $e^{A\tau} = I + A\tau + A^2\tau^2/2$, and the columns of $e^{A\tau}B$ are read off the graph as walk sums (Example 3.7):

$$e^{A\tau}e_1 = \begin{bmatrix} 1 \\ \tau \\ \tau \\ \tau^2 \end{bmatrix}, \quad e^{A\tau}e_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \tau \end{bmatrix}, \quad e^{A\tau}e_4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

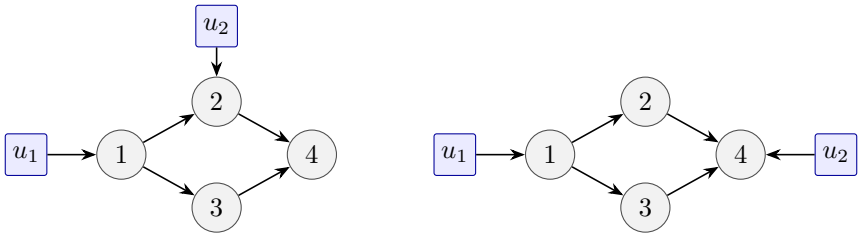
Integrating the outer products over $\tau \in [0, 1]$ gives, exactly,

$$W_{\{1,2\}} = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{2} & \frac{1}{3} \\ \frac{1}{2} & \frac{4}{3} & \frac{1}{3} & \frac{3}{4} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{3} & \frac{3}{4} & \frac{1}{4} & \frac{8}{15} \end{bmatrix}, \qquad W_{\{1,4\}} = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{2} & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{4} & \frac{6}{5} \end{bmatrix}.$$

The second and third rows of $W_{\{1,4\}}$ are *identical* (the twins again, visible in the matrix itself), so $e_2 - e_3$ lies in its kernel. The metric values:

metric at $T = 1$	$\mathcal{D} = \{1, 2\}$	$\mathcal{D} = \{1, 4\}$
$\mathrm{tr}(W)$	$16/5 = 3.20$	$43/15 \approx 2.87$
$\lambda_{\min}(W)$	≈ 0.043	0
$\det W$	$1/135$	0
$\log\det W$	≈ -4.91	$-\infty$
energy to reach $e_2 - e_3$	16	∞

By trace the two placements look nearly interchangeable (3.20 versus 2.87), but by the other two metrics they are not even in the same class. The structural chapters explain why. With drivers $\{1, 2\}$ the maximum minor built from $[\, e_1,\ e_2,\ Ae_1,\ Ae_2 \,]$ equals $a_{31}a_{42}$ (a single monomial, non-zero for every realization), so $\{1, 2\}$ is not just controllable here but SSC (sufficiency by the single-monomial argument of Chapter 5, whose multi-input section notes that for several inputs one never-vanishing minor is a sufficient condition). With drivers $\{1, 4\}$ the controllable directions are spanned by e_1, e_4, and the *single* combination $a_{21}e_2 + a_{31}e_3$: rank three for every realization. This is exactly the augmented pair of Example 8.9, where node 4 passed the FSC test, meaning a driver at 4 does not increase the generic dimension. The budget is spent on a node the placement $\{1\}$ already covered, and the effective input sites (Section 8.6) are the non-fixed twins. The Gramian turns that structural conclusion into energy: splitting the twins costs 16 under $\{1, 2\}$ and cannot be achieved at any finite energy under $\{1, 4\}$.



(a) $\mathcal{D} = \{1, 2\}$ (b) $\mathcal{D} = \{1, 4\}$

Figure 10.1: The two placements compared in Example 10.2, both of size two, on the diamond of Fig. 6.1. (a) $\mathcal{D} = \{1, 2\}$ splits the twins $2, 3$. (b) $\mathcal{D} = \{1, 4\}$ spends its second input on node 4, already an FSC node of the one-input diamond (Example 8.9), so that input raises no dimension. The metric table of Example 10.2 scores the two at $T = 1$: $\mathrm{tr}(W)$ reads $16/5$ against $43/15$, a gap the trace alone would call small, while $\lambda_{\min}(W)$ reads 0.043 against 0 and $\det W$ reads $1/135$ against 0. Under (b) the direction $e_2 - e_3$ lies in $\ker W_{\mathcal{D}}(1)$ and cannot be reached at any finite energy.

The example generalizes into the working rules of the field: $\mathrm{tr}(W)$ has the lowest computational cost and is the best behaved for optimization (next section), but it guarantees nothing. An uncontrollable placement can score near the top, because an average is dominated by the easy directions. $\lambda_{\min}(W)$ guarantees everything but is the hardest to optimize, and in large networks with few drivers it is typically minuscule: worst-case control energy grows exponentially, under broad assumptions, as the driven fraction of nodes shrinks [36], and for targeted control it grows with the graph distance between drivers and targets [37]. $\log\det W$ sits usefully in between: sensitive to every direction, finite only for controllable placements, and blessed with the combinatorial structure the next section exploits.

10.2 Optimal Placement Problem

The design problem can now be stated. It is the point of the book where the two halves (numerical Part I, structural Parts II–III) finally recombine: the *pattern* decides which placements are feasible at all, the *numbers* decide which feasible placement is best.

Problem 10.1 (Optimal input placement problem)**.** Given a pattern A on the graph $\mathcal{G}$, a budget k, a horizon T, and a metric $f \in \{\mathrm{tr}, \lambda_{\min}, \log\det\}$, choose a driver set $\mathcal{D} \subseteq \mathcal{V}^S$ with $|\mathcal{D}| \le k$ (and, in the gain-design variant, the non-zero entries of a B supported on $\mathcal{D}$) maximizing $f(W_{\mathcal{D}}(T))$ subject to the family $\mathcal{Q}(A)$ with $B_{\mathcal{D}}$ being SC (respectively SSC).

Three comments on the formulation. First, the feasibility constraint is the whole of Chapters 7 and 9 re-read as a constraint set: SC feasibility is a matching condition (Theorem 7.2), polynomially checkable, while SSC feasibility is a zero-forcing condition (Theorem 7.14, under its free-self-loop hypothesis), polynomial to check for a given $\mathcal{D}$, though minimizing over $\mathcal{D}$ is NP-hard (Remark 9.7). Second, the metric uses a *realization*: in a structured network one evaluates f at a nominal realization, or averages it, or takes the worst case over the family $\mathcal{Q}(A)$. The last choice is the true structural analogue and is essentially unexplored (see Section 10.4). Third, cost need not be cardinality: attaching an input to node i may carry a cost c_i, and the budget $|\mathcal{D}| \le k$ becomes $\sum_{i\in\mathcal{D}} c_i \le c$. For purely *structural* objectives of that kind (feasibility at minimum cost, no Gramian), the selection problem is solvable in polynomial time by the framework of [38], which is worth knowing before reaching for heuristics.

However, with a Gramian in the objective, the problem is genuinely combinatorial: even with all weights known exactly, selecting a minimal set of standard-basis input columns rendering (A, B) controllable is NP-hard [33], and metric optimization inherits the obstruction. The practical theory is built on one algebraic property. Because the placement matrix satisfies $B_{\mathcal{D}}B_{\mathcal{D}}^{\top} = \sum_{i\in\mathcal{D}} e_i e_i^{\top}$, the Gramian is *additive over drivers*:

$$W_{\mathcal{D}}(T) \;=\; \sum_{i\in\mathcal{D}} W_{\{i\}}(T), \tag{10.2}$$

one fixed positive-semidefinite increment per node, independent of the rest of $\mathcal{D}$. Consequences, at statement level, with proofs in the cited literature:

- *Trace is modular:* by Eq. (10.2), $\mathrm{tr}\, W_{\mathcal{D}} = \sum_{i\in\mathcal{D}} \mathrm{tr}\, W_{\{i\}}$, so the trace-optimal budget-k set is simply the k nodes with the largest individual scores, and no search is needed. This is the metric behind the "average controllability" rankings of network neuroscience [35].
- *Log-determinant is submodular:* $\mathcal{D} \mapsto \log\det(W_{\mathcal{D}} + \varepsilon I)$ (regularized so that infeasible sets score $\approx \log\varepsilon$ rather than $-\infty$) is monotone and submodular (adding a driver helps less the richer the set already is), so greedy selection, adding at each step the node with the largest marginal gain, is guaranteed a $(1 - 1/e)$-approximation of the optimum [39]. The ratio is understood for the normalized gain $g(\mathcal{D}) = \log\det(W_{\mathcal{D}} + \varepsilon I) - n\log\varepsilon$, which vanishes at $\mathcal{D} = \varnothing$.
- *The metric that guarantees controllability is the hardest to optimize:* $\lambda_{\min}(W_{\mathcal{D}})$ is neither modular nor submodular, and greedy can fail it badly. Guarantees exist only in weakened or averaged forms [39, 36]. Structured heuristics (choose among the *feasible* minimum placements of Chapter 9 by a Gramian score) are the common compromise, and joint metrics mixing controllability with robustness are treated in the leader-selection literature [40].

Example 10.3 (Greedy on the diamond)**.** The single-driver traces at $T = 1$, by Eq. (10.2) and the columns computed in Example 10.2: nodes $1, 2, 3, 4$ score $\frac{28}{15}$, $\frac{4}{3}$, $\frac{4}{3}$, 1. Greedy (equivalently, top-k, since trace is modular) picks node 1, then a twin: $\mathcal{D} = \{1, 2\}$, the placement that Example 10.2 showed to be SSC. Here the low-cost metric and the structural conclusion happen to agree, and the additivity check $\frac{28}{15} + \frac{20}{15} = \frac{16}{5}$ confirms the table. The agreement is a coincidence, not a rule: reweight the diamond with $a_{42} = a_{43} = 3$ and the twins' scores rise to 4 each, so the trace-optimal pair becomes $\{2, 3\}$, which leaves node 1 unreachable and is not even SC. Feasibility first, metric second: with that discipline the trace is a useful low-cost metric. Without it, the trace selects infeasible placements.

10.3 Everything Dualizes

This book developed one body of results, and duality (Theorem 3.11) doubles it. Transpose the system and every object of Parts II–III maps to a sensing dual: drivers become output nodes, the input-walks of Chapter 4 become walks *into* those nodes (stems become branches), SC and SSC become their observability counterparts, the subspaces of Chapter 6 become statements about the kernel of $\mathcal{O}$, the node-level theory of Chapter 8 classifies which *states can be reconstructed* at every realization [31, 1], and the minimum input problem of Chapter 9 becomes minimum output placement on the edge-reversed graph. The Gramian results transfer verbatim: the observability Gramian $W_o(T) = \int_0^T e^{A^\top \tau} C^\top C e^{A\tau}\, d\tau$ measures the output energy $x_0^\top W_o(T)\, x_0$ that an initial state deposits on the output nodes, its small eigenvalues flag the directions hardest to *see*, and Problem 10.1 with $(A, B) \mapsto (A^\top, C^\top)$ is the optimal output placement problem, greedy guarantees included. Every result of this book is, silently, two results. The reader who reproves any of them in terms of sensing will find nothing new and understand everything better.

10.4 Open Problems and Course Projects

The field closes no chapter cleanly, and this book should not pretend otherwise. Four problems, each open as of this writing [4, 1], each scoped so that a semester of work produces either partial results or an instructive negative answer.

Project 1: mixed weights. Part II's dichotomy (every weight either exactly zero or freely non-zero) is a modeling idealization. Real networks are *partially structured*: some entries are fixed numbers (a circuit's fixed resistor, a known line impedance), others free (Remark 2.11 already noted that dependent entries need care). Fixing some weights shrinks the family $\mathcal{Q}(A)$, so the answers can only improve: patterns that are SC but not SSC may become controllable for the whole reduced family. *Project*: redo the trichotomy of Chapter 5 for mixed patterns, starting with the flip family of Example 3.13 with a_{11} held at one numerical value (for which values does the family become uniformly controllable?), then attempt a mixed-weight version of one graph criterion from Chapter 7.

Project 2: the exact worst-case dimension beyond HDAGs. $\dim \mathcal{C}_\Gamma$ has a tight shortest-stem lower bound and an exact algorithm only on hierarchical DAGs (Section 6.4). The dissertation conjectures the same layer logic extends to general DAGs [1]. *Project*: enumerate all single-input DAG patterns on up to six nodes, compute $\dim \mathcal{C}_\Gamma$ symbolically

(one computer-algebra run per graph class, in the style of Chapter 5), and map exactly where the shortest-stem bound loses tightness once walks of different lengths reach the same node. For instance, on the diamond of Fig. 6.1 with the extra edge $1 \to 4$, node 4 is reached in two steps and in three. A proof for any class strictly beyond HDAGs would be publishable.

Project 3: FSSC nodes on general graphs. The node-level worst-case theory of Chapter 8 is effective exactly where $\dim \mathcal{C}_\Gamma$ is, and even there only in one direction: the FSSC half of Theorem 8.7 rules a node out when the added input raises $\dim \mathcal{C}_\Gamma$, but decides nothing when it does not, and $\dim \mathcal{C}_\Gamma$ itself is computable only on HDAGs, open in general. The known necessary and sufficient condition for FSSC nodes on general graphs is the subspace equality of Lemma B.3, which is state-space, not graph-theoretical [12, 4]. For SSC nodes the subset condition of Definition 8.16 is graph-theoretical but exhaustive, and a polynomial criterion is known only on polytrees (Theorem 8.17). *Project*: extend the shortest-path-graph condition of Theorem 8.17 to one family with cycles (start: a single cycle with one entry point), or build a counterexample showing why that condition fails once walk lengths mix. Either outcome is a result.

Project 4: energy-aware placement under structural constraints. Part II's answers are binary, whereas energy is continuous. A pattern can be SC while a realization sits arbitrarily close to a cancellation (on the diamond, let $a_{42}a_{21} + a_{43}a_{31} \to 0$ and the energy to reach e_4 diverges), and even an SSC chain becomes practically uncontrollable as a weight tends to zero: the dissertation calls the distance to the nearest failing realization a *control margin* [1]. The true structural metric is therefore worst-case over the family, $\inf_{A \in \mathcal{Q}(A)} \lambda_{\min}(W(T))$ under a normalization of the weights, a quantity about which almost nothing is known. *Project*: compute it numerically on the running example graphs of this book as the weights range over a box, locate the minimizing realizations relative to the cancellation loci of Chapter 5, and formulate (or refute) the natural conjecture that margins are governed by the same multi-terms that governed rank [36, 37].

The book ends where it began, and that is the point. One object (the determinant of the controllability matrix, a polynomial whose monomials are disjoint sets of stems and cycles) carried the whole development: its non-vanishing was controllability (Chapter 3), its generic and worst-case behavior split SC from SSC (Chapter 5), its surviving minors measured subspaces (Chapter 6) and nodes (Chapter 8), its combinatorics became graph criteria (Chapter 7) and placement rules (Chapter 9), and its magnitude (through the Gramian) measured the placements this chapter compared. The open problems above are the places where that object remains open. Take one.

Sources. The metric interpretations and their limitations follow [36]. Submodularity and greedy guarantees are from [39], hardness from [33], the structural cost framework from [38], leader selection from [40], and the energy-scaling and application context from [37, 35]. The open problems are drawn from the survey's concluding section [4] and the dissertation's closing chapter [1], with the node-level state of the art in [12, 31].

Exercises

Exercise 10.1 (Three metrics, one formula)**.** Give the exact reading of $\lambda_{\min}(W)$, $\operatorname{tr}(W^{-1})$ and $\log\det W$ in terms of the minimum-energy formula Eq. (10.1). Which two of $\operatorname{tr}(W)$, $\lambda_{\min}(W)$

and $\log\det W$ take a degenerate value exactly at the uncontrollable placements (Section 10.1), and which one guarantees nothing?

Exercise 10.2 (Energy for four targets)**.** Using $W_{\{1,2\}}(1)$ of Example 10.2, compute the minimum energy Eq. (10.1) for each of the targets e_1, e_2, e_3, e_4. Why is e_3 so much more expensive than e_2, although the twins play symmetric roles in the pattern? [Check: $E(e_2 - e_3) = 16$, as the table reports.]

Exercise 10.3 (Additivity over drivers)**.** Compute $W_{\{4\}}(1)$ for the diamond with unit weights and use it to verify Eq. (10.2) on the placement $\mathcal{D} = \{1, 4\}$: the single-driver traces of Example 10.3 must add up to the 43/15 tabulated in Example 10.2.

Exercise 10.4 (A trace-optimal pair that fails)**.** Reweight the diamond with $a_{42} = a_{43} = 3$ and all other weights 1. Compute the four single-driver traces $\operatorname{tr} W_{\{i\}}(1)$, confirm the claim of Example 10.3 that the trace-optimal pair is then $\{2, 3\}$, and show that this pair is not even SC.

Exercise 10.5 (Modular against submodular)**.** Explain why the trace-optimal driver set of size k needs no search at all, why the same argument fails for $\log\det W$, and what submodularity guarantees for greedy selection instead (Section 10.2).

Exercise 10.6 (Minimum outputs by duality)**.** Read Theorem 9.1 through the duality of Section 10.3: reverse every edge of the four-node network and apply the theorem to the result. How many output nodes does the observability counterpart require, and where must they attach? Compare with the single output $y = x_4$ of Example 2.4 and its determinant in Example 3.12.

Appendix A

Proofs: Structural Side

The non-trivial proofs behind the SC column of the book's grid, collected here so the main text can move without interruption. Each section restates a result of Part II by cross-reference, proves it in the book's notation, and closes with a source-and-notation note recording the origin and the notation translation performed. Trivial or two-line arguments stay where they were stated, and only proofs that are genuinely long appear below.

A.1 Determinant Trichotomy and Genericity Principle

We prove Proposition 5.4: for a single-input pattern ($m = 1$) with square $\mathcal{C}_F$, exactly one of the three cases holds: (i) $\det \mathcal{C}_F \equiv 0$ (never controllable), (ii) $\det \mathcal{C}_F \not\equiv 0$ but vanishing for some realization (SC, with a non-empty measure-zero uncontrollable set), and (iii) $\det \mathcal{C}_F$ nowhere zero (SSC). All the content is in case (ii). Cases (i) and (iii) are immediate from the definitions. Case (ii) rests on a classical fact, recorded first as a lemma.

Lemma A.1 (Genericity)**.** *A polynomial π in r real variables that is not identically zero vanishes on a set of Lebesgue measure zero in $\mathbb{R}^r$.*

Proof. Induction on r. For $r = 1$ a non-zero univariate polynomial has finitely many roots, a set of measure zero. For the step, write

$$\pi(a_1, \ldots, a_r) = \sum_{j=0}^{d} c_j(a_1, \ldots, a_{r-1})\, a_r^j.$$

Since $\pi \not\equiv 0$, some coefficient $c_j \not\equiv 0$. By the inductive hypothesis its zero set $Z \subseteq \mathbb{R}^{r-1}$ has measure zero. For every fixed $a' = (a_1, \ldots, a_{r-1}) \notin Z$ the univariate polynomial $a_r \mapsto \pi(a', a_r)$ is non-zero, hence has finitely many roots. The zero set of π therefore meets each line $\{a'\} \times \mathbb{R}$ with $a' \notin Z$ in finitely many points. By Fubini's theorem its r-dimensional measure is zero. □

Proof of Proposition 5.4. Let $m = 1$, so $\mathcal{C}_F$ is $n \times n$ and $\pi(a) := \det \mathcal{C}_F$ is a polynomial in the network parameters $a = (a_1, \ldots, a_r)$, the structurally non-zero entries of A_F. The realizations of the pattern (Definition 4.1) are exactly the points of the open set $D = \{a \in \mathbb{R}^r : a_i \neq 0 \ \forall i\}$, whose complement (a finite union of coordinate hyperplanes) has measure zero.

The three cases are exhaustive and mutually exclusive: either $\pi \equiv 0$, or $\pi \not\equiv 0$ and then either π has no zero in D or it has one. It remains to identify each with the stated outcome.

Case (i). If $\pi \equiv 0$ then $\det \mathcal{C}_F = 0$ for every $a \in D$, so $\operatorname{rank} \mathcal{C}_F < n$ at every realization. By the Kalman rank condition (Theorem 3.2) no realization is controllable, and in particular the family is not SC.

Case (iii). If π has no zero in D then $\det \mathcal{C}_F \neq 0$ for *every* $a \in D$, so every realization is controllable: the family is SSC by Definition 5.2.

Case (ii). Suppose $\pi \not\equiv 0$ yet $\pi(a^\circ) = 0$ for some $a^\circ \in D$. By Lemma A.1 the zero set $Z(\pi)$ has measure zero, so $Z(\pi) \cap D$ has measure zero as well. On the full-measure remainder $D \setminus Z(\pi)$ every realization is controllable, so the family is SC (Definition 5.1), and the uncontrollable realizations form the set $Z(\pi) \cap D$, non-empty because a° lies in it and of measure zero by the preceding sentence. Note the constructive reading: because $D \setminus Z(\pi)$ is non-empty, *one* controllable realization already establishes SC. Exhibiting a single realization with $\det \mathcal{C}_F \neq 0$ suffices, and genericity does the rest. □

Source and notation. Adapted from the definitions of SC and SSC and the worked three-node example of [1]. The genericity principle (a non-trivial polynomial vanishes only on a measure-zero variety) is classical and stated for verification against [10, 18]. Notation translated as: the source's controllability matrix $\mathcal{C}_{\mathcal{F}}$ is the book's $\mathcal{C}_F$, and the source's verbal "controllable for almost all / for all network parameters" are Definitions 5.1 and 5.2.

A.2 Generic Dimension: Hosoe's Cover Theorem

We prove Theorem 6.4: $\dim \mathcal{C}_\Lambda$ equals the maximum number of state nodes that can be covered by a node-disjoint set of stems and cycles on the reachable part of the pattern graph. Write g for that cover count, computed on the subgraph $\mathcal{G}_{\text{reach}}$ induced by the reachable state nodes (Definition 4.13) together with the input nodes. The claim is $\dim \mathcal{C}_\Lambda = g$.

Proof of Theorem 6.4. By definition $\dim \mathcal{C}_\Lambda = \max_{A_F \in \mathcal{Q}(A)} \operatorname{rank} \mathcal{C}_F$. Each minor of $\mathcal{C}_F$ is a polynomial in the parameters, so by Lemma A.1 it is either identically zero or non-zero on a full-measure set. There are finitely many minors, so on the common full-measure intersection every non-identically-zero minor is simultaneously non-zero. Hence

$$\dim \mathcal{C}_\Lambda \;=\; \rho \;:=\; \max\{\, r : \text{some } r \times r \text{ minor of } \mathcal{C}_F \text{ is } \not\equiv 0 \,\}.$$

We show $\rho = g$ by two inequalities.

Lower bound $\rho \geq g$ (construction). Let $\mathcal{H}$ be a node-disjoint set of stems and cycles covering g reachable state nodes. To each covered node l attach a walk w_l reaching it along $\mathcal{H}$: if l lies on a stem, w_l is the stem prefix up to l. If l lies on a cycle, w_l is a shortest feeding stem into that cycle (which exists, since l is reachable) continued around the cycle to l, going round extra times if we wish. Write u_l for the input rooting w_l and j_l for its number of steps counted from the driver node, as in Proposition 4.24. Choose the walks so that no two nodes fed by the same input carry the same j_l: the nodes of one stem sit at distinct step counts, and a cycle node's step count may be raised by the cycle length as often as needed. Form the $g \times g$ submatrix M of $\mathcal{C}_F$ with rows the covered nodes and, for each l, the column $A_F^{j_l} b_{u_l}$, where b_{u_l} is the column of B carrying the input u_l. The columns are then pairwise distinct. A step count may exceed $n-1$, beyond the columns $\mathcal{C}_F$ carries. That is harmless, since by the Cayley–Hamilton theorem $\operatorname{Im}[\, B, A_F B, \ldots, A_F^{j} B \,] = \operatorname{Im} \mathcal{C}_F$ for every j, so a non-zero $g \times g$ minor built from such columns still forces $\operatorname{rank} \mathcal{C}_F \geq g$ wherever it does not vanish.

By Proposition 4.24 the $(l, \text{column } l)$ entry of M contains the WP of w_l. Because the stems and cycles of $\mathcal{H}$ are node-disjoint, the monomial obtained by multiplying these g WPs uses each edge of $\bigcup_l w_l$ a determined number of times, and the only assignment of columns to rows that reproduces it is the diagonal one. Any other bijection would need a walk through a node already committed elsewhere, or one repeating a committed step count, and would alter the monomial. That monomial therefore survives in the Leibniz expansion of $\det M$ with no canceling term, so $\det M \not\equiv 0$ and $\rho \geq g$. The minor itself need not reduce to that one monomial: on the pattern $1\to2$, $2\to2$, $1\to3$, $3\to3$ driven at node 1, the cover made of the stem $u\to1$ and the two self-loops gives $\det M = a_{21}a_{31}(a_{33} - a_{22})$. What disjointness gives is not a lone monomial but a monomial with no partner to cancel it.

Upper bound $\rho \leq g$. This half is Hosoe's own, resting on the generic-rank analysis of structured matrices of [18]. We quote it here rather than reproduce it. What it says in the present notation is that no realization can beat the best disjoint stem-and-cycle cover: a non-zero $r \times r$ minor of $\mathcal{C}_F$ selects r distinct rows and, for each, a walk from an input node whose step count is the one its column carries. A maximum such selection reduces to stems whose revisited portions close into node-disjoint cycles, a disjoint stem-and-cycle cover of ρ nodes, each of them reachable because it lies on an input-rooted walk. Hence $\rho \leq g$.

Combining, $\dim \mathcal{C}_\Lambda = \rho = g$. The restriction to $\mathcal{G}_{\text{reach}}$ is essential in the lower bound: a covered cycle disconnected from every input has identically-zero rows in $\mathcal{C}_F$ (Proposition 4.24) and contributes no surviving monomial, exactly the case isolated in Example 7.4. □

Source and notation. The theorem is Hosoe's [19] (verified only, adapted into the book's notation). The lower bound is proved above, while the upper bound is quoted from that source, whose generic-rank analysis of structured matrices rests on Poljak's [18]. Notation translated as: the source's $|\mathcal{C}_\Lambda|$, the dimension of SCS on $\mathcal{G}(T_c)$, is the book's $\dim \mathcal{C}_\Lambda$ on the pattern graph. The source's "influenceable" restriction is the book's input-connectedness (Definition 4.13), and Λ denotes the generic (maximum) case as in the book.

A.3 Lin's Theorem: Sufficiency and Dilation Equivalence

Theorem 7.2 states three equivalent conditions for a pattern graph with at least one input: (i) SC, (ii) input-connected and some disjoint set of stems and cycles covers all state nodes, and (iii) input-connected and dilation-free. The main text (Chapter 7) already gives the two necessity arguments: an unreachable node and a dilation each force a rank drop in every realization. Two implications remain: the graph equivalence (ii)⇔(iii) and the sufficiency (ii)⇒(i).

Proof that (ii)⇔(iii). Both conditions include input-connectedness, so it suffices to prove, on any pattern, that a disjoint set of stems and cycles covers all state nodes if and only if the pattern is dilation-free. Form the bipartite graph H with the state nodes $\mathcal{V}^S$ on the right, all nodes (state or input) on the left, and an edge p–s whenever p feeds s in the pattern graph, i.e. $p \in N^{\text{in}}(\{s\})$ (Definition 7.1). A matching of H saturating every right node assigns each state node a *distinct* in-neighbor. By Hall's marriage theorem [7] such a saturating matching exists if and only if $|N^{\text{in}}(S)| \geq |S|$ for every non-empty $S \subseteq \mathcal{V}^S$, that is, if and only if no dilation exists.

It remains to identify saturating matchings with covers. Given a saturating matching, read its edges in the pattern graph: every state node is a head exactly once and every node is a

tail at most once, so the matched edges form node-disjoint paths and cycles covering all state nodes. A path can begin only at a node that is never a head, and since every state node is matched as a head, each path begins at an input and is therefore a stem. Thus the matching is a disjoint stem-and-cycle cover of $\mathcal{V}^S$. Conversely, in any such cover each state node has a unique in-neighbor along its stem or cycle, and these in-edges form a matching saturating $\mathcal{V}^S$. The two notions coincide, completing the equivalence. □

Proof that (ii)⇒(i). Assume every state node reachable and fix a disjoint set of stems and cycles $\mathcal{H}$ covering all state nodes. First convert $\mathcal{H}$ into *spanning cacti*. Each cycle of $\mathcal{H}$ is reachable, so choose a shortest walk from an input to it and attach the cycle to the already-built stem structure through the single edge by which that walk first enters the cycle, the *bridge edge*. A cycle together with its bridge edge is a *bud*, and performing this for every cycle yields node-disjoint cacti (stems carrying buds) that together span $\mathcal{V}^S$, one cactus per input used.

By Lemma A.1 it now suffices to exhibit *one* realization at which some $n \times n$ minor of $\mathcal{C}_F$ is non-zero: each such minor is a polynomial in the parameters, so one that is non-zero somewhere is non-zero off a measure-zero set, leaving $\operatorname{rank} \mathcal{C}_F = n$ for almost all realizations, which is SC. Give value 1 to every stem edge and every bridge edge of the cacti, choose the cycle weights of the buds as in Lin's construction (the choice matters: with equal cycle weights two self-loop buds bridged from the same stem node contribute the same direction at every step), and value $\epsilon \neq 0$ to every remaining structurally non-zero entry. As $\epsilon \to 0$ the pair block-decomposes over the node-disjoint cacti. Within one cactus, writing b for the column of B of the input that roots it, the columns $b, A_F b, A_F^2 b, \ldots$ reach the stem nodes in order (a lower-triangular full-rank block on the stem), and each bud, entered through its bridge edge at one stem node, is then traversed one cycle step at a time. That the cycle weights can be chosen so that the buds contribute as many further independent directions as their cycles have nodes is Lin's step, quoted here rather than reproved. Collecting one such block per cactus gives an $n \times n$ minor of $\mathcal{C}_F$ that is non-zero at $\epsilon = 0$. That minor is a polynomial in ϵ, hence non-zero for all sufficiently small $\epsilon \neq 0$, which gives a realization of full rank, and by Lemma A.1 the family is SC. □

Source and notation. The equivalence (ii)⇔(iii) is Hall's theorem applied to the bipartite graph representation [7]. The sufficiency is Lin's cacti construction [16], in the multi-input form of [22] (both third-party, rewritten in the book's notation, with the step that makes a bud's cycle contribute independent directions quoted rather than reproved). Notation translated as: the survey [4] names this notion "accessible" and defines "input-connected" differently, though its theorems use that word in the sense printed here (Definition 4.13), the sense of the author's later papers. The survey states only the cover form (ii), while the dilation form (iii) is the book/Poljak reading via $N^{\text{in}}(S)$ (Definition 7.1). SC is established from a single realization of full rank by Lemma A.1, which covers the multi-input case that Proposition 5.4 does not.

A.4 Fixed-Node Input-Addition Test: Max-Dimension Side

We prove the FSC half of Theorem 8.7: on an input-connected pattern, node k is an FSC node if and only if adding a dedicated input at k (replacing B by $B_k = [\,B,\ e_k\,]$) does not increase $\dim \mathcal{C}_\Lambda$. (The SSC-side clause of that theorem, with $\dim \mathcal{C}_\Gamma$, is only necessary, not sufficient, and it is proved in the companion Appendix B, where Lemma B.3 supplies the exact criterion.) Throughout, $g := \dim \mathcal{C}_\Lambda$ of the pair (A_F, B) and $g' := \dim \mathcal{C}_\Lambda$ of the augmented

pair (A_F, B_k). Write $\mathcal{C}_{F,k}$ for the controllability matrix of the augmented pair. Since $\mathcal{C}_{F,k}$ contains the columns of $\mathcal{C}_F$, always $g' \geq g$.

The argument rests on the A_F-invariance of the controllable subspace. For a fixed realization,

$$\operatorname{Im}\mathcal{C}_{F,k} \;=\; \operatorname{Im}\mathcal{C}_F \;+\; \langle A_F \mid e_k\rangle, \qquad \langle A_F \mid e_k\rangle := \operatorname{span}\{e_k, A_F e_k, \ldots, A_F^{n-1} e_k\}, \tag{A.1}$$

because $\operatorname{Im}\mathcal{C}_{F,k}$ is the smallest A_F-invariant subspace containing both $\operatorname{Im} B$ and e_k, and $\operatorname{Im}\mathcal{C}_F$ is already the smallest one containing $\operatorname{Im} B$.

Proof of Theorem 8.7, FSC part. Recall $\mathcal{C}_\Lambda^F = \bigcap\{\operatorname{Im}\mathcal{C}_F : \operatorname{rank}\mathcal{C}_F = g\}$, the intersection over generic (maximum-dimension) realizations, and that k is an FSC node iff $e_k \in \mathcal{C}_\Lambda^F$ (Definitions 8.2 and 8.6).

For the *only if* condition, suppose that k is an FSC node. By Lemma A.1 the parameters at which *both* $\operatorname{rank}\mathcal{C}_F = g$ and $\operatorname{rank}\mathcal{C}_{F,k} = g'$ are attained form a full-measure set. Pick a realization $A_F^\star$ in it. It is a maximum-dimension realization of the original pair, so $e_k \in \operatorname{Im}\mathcal{C}_F^\star$ because $e_k \in \mathcal{C}_\Lambda^F$. As $\operatorname{Im}\mathcal{C}_F^\star$ is $A_F^\star$-invariant, $A_F^{\star j} e_k \in \operatorname{Im}\mathcal{C}_F^\star$ for all j, whence $\langle A_F^\star \mid e_k\rangle \subseteq \operatorname{Im}\mathcal{C}_F^\star$ and, by (A.1), $\operatorname{Im}\mathcal{C}_{F,k}^\star = \operatorname{Im}\mathcal{C}_F^\star$. Taking dimensions, $g' = \operatorname{rank}\mathcal{C}_{F,k}^\star = \operatorname{rank}\mathcal{C}_F^\star = g$: the addition does not increase $\dim\mathcal{C}_\Lambda$.

For the *if* condition, we prove the contrapositive: suppose k is not an FSC node, i.e. $e_k \notin \mathcal{C}_\Lambda^F$. Then some maximum-dimension realization $A_F^\star$ (with $\operatorname{rank}\mathcal{C}_F^\star = g$) has $e_k \notin \operatorname{Im}\mathcal{C}_F^\star$. By (A.1), $\operatorname{Im}\mathcal{C}_{F,k}^\star$ strictly contains $\operatorname{Im}\mathcal{C}_F^\star$, so $\operatorname{rank}\mathcal{C}_{F,k}^\star \geq g+1$. Hence $g' = \max\operatorname{rank}\mathcal{C}_{F,k} \geq g+1 > g$: the addition increases $\dim\mathcal{C}_\Lambda$.

The two implications together give: k is an FSC node iff adding a dedicated input at k leaves $\dim\mathcal{C}_\Lambda$ unchanged. □

Source and notation. The fixed subspace and this max-dimension input-addition test originate with Commault, van der Woude and Boukhobza [29]. The argument above is the author's restatement [1], matching the survey's SC-node theorem [4]. Notation translated as: the source's "leader" is the book's driver node, the augmented input is $B_k = [\,B,\ e_k\,]$ with e_k the standard basis vector, and the source's "dimension of SCS" is the book's $\dim\mathcal{C}_\Lambda$. The equality (A.1) is the invariance argument sketched after Theorem 8.7.

A.5 Matched-Node Criterion on Hierarchical DAGs

We prove Theorem 8.10: on an HDAG (Definition 4.21) whose drivers are its source nodes, so that the recursive layers are the distance layers, a node $i \in \ell_k$ is an FSC node if and only if i is matched in *every* feasible maximum disjoint-stem set for ℓ_k. Since the pattern is a DAG, Theorem 6.4 reduces $\dim\mathcal{C}_\Lambda$ to the maximum number of state nodes covered by disjoint *stems* (no cycles), and Theorem 8.7 reduces FSC-node membership to the non-increase of that cover count under a new driver. One structural observation links the whole-graph cover to the per-layer sets.

Lemma A.2 (Layerwise decomposition)**.** *In an HDAG (Definition 4.21) whose drivers occupy ℓ_1, so that the layers are the distance layers, every stem meets each layer in at most one node, and a node-disjoint set of stems covers the maximum number of state nodes overall if and only if, for each layer ℓ_k, it covers a maximum number of nodes of ℓ_k attainable by disjoint stems.*

Proof. No edge skips a layer. On an HDAG every walk into a node has one common length, and ℓ_r collects the nodes reachable from the inputs in exactly r steps (Definition 4.21). So if an edge runs from a node of ℓ_r to a node v, then v is reachable in $r+1$ steps, and every walk into v has that length: $v \in \ell_{r+1}$. A stem rooted in ℓ_1 therefore meets $\ell_1, \ell_2, \dots$ in exactly one node each until it stops, reaching ℓ_k at its k-th step, and meets no layer twice.

Write $c_k(\mathcal{S})$ for the number of nodes of ℓ_k that a node-disjoint set of stems $\mathcal{S}$ covers (equivalently, the number of its stems that reach ℓ_k), and put $M_k := \max_{\mathcal{S}} c_k(\mathcal{S})$. The layers partition the state nodes, so $\mathcal{S}$ covers $\sum_k c_k(\mathcal{S})$ nodes in all, with $c_k(\mathcal{S}) \le M_k$ term by term.

For the *if* condition, suppose $c_k(\mathcal{S}) = M_k$ in every layer. Then $\mathcal{S}$ covers $\sum_k M_k$ nodes, a total no set can exceed, so it covers as many state nodes as possible. The *only if* condition follows once that total is known to be attained, since a set reaching it while obeying $c_k \le M_k$ must have $c_k = M_k$ in every layer. Two steps give the attainment.

Exchange step. If $q \le M_k$ node-disjoint stems reach ℓ_k and end at a set X of q nodes there, some maximum disjoint-stem set for ℓ_k matches every node of X. Delete the layers below ℓ_k, those of larger index. In what is left no edge leaves ℓ_k. Compare the q stems with a set of M_k disjoint stems reaching ℓ_k and apply the standard exchange argument for systems of node-disjoint paths: it raises the count to M_k one stem at a time, re-drawing existing stems as it goes. No node of X is released on the way, because a re-drawn stem can enter ℓ_k only at its last step (no edge lets it leave again), so a node of ℓ_k that already carries the end of a stem keeps one. The ends grow from X to a set of M_k nodes containing X.

Attainment. What a set covers below ℓ_k depends only on which nodes of ℓ_k it uses, not on how they were reached, so its stems above ℓ_k may be re-drawn freely as long as those nodes are kept. Descend from the last layer ℓ_p: start with M_p disjoint stems reaching it. Given M_{k+1} disjoint stems that reach ℓ_{k+1} and attain M_j in every layer ℓ_j with $j > k$, let $X \subseteq \ell_k$ be the nodes of ℓ_k they use. Each of those stems meets ℓ_k, so $|X| = M_{k+1}$. The exchange step replaces their parts above ℓ_k by M_k disjoint stems whose ends include X, and re-attaching below every node of X the part that continued below it leaves the deeper layers as they were. The new set attains M_j in every layer with $j \ge k$. At $k = 1$ the set covers $\sum_k M_k$ nodes in all. □

Remark A.3 (Layer maxima need not add up). Without the hierarchy the *only if* condition fails. On $1 \to 3$, $1 \to 4$, $2 \to 3$, $2 \to 4$, $2 \to 6$, $3 \to 5$, $3 \to 6$ with drivers $1, 2$, recursion gives $\{1,2\}, \{3,4\}, \{5,6\}$ and each layer admits two covered nodes, yet no disjoint set of stems covers more than five of the six: the edge $2 \to 6$ skips a layer, so the per-layer maxima $2+2+2$ are not attainable together.

For a layer ℓ_k let the feasible maximum disjoint-stem sets be $\mathcal{M}^{(1)}(\ell_k), \dots, \mathcal{M}^{(n_k)}(\ell_k)$ and let $\widehat{\mathcal{M}}^{(t)}(\ell_k)$ denote the set of nodes of ℓ_k that $\mathcal{M}^{(t)}(\ell_k)$ matches (the node of ℓ_k lying on one of its stems, at most one per stem by Lemma A.2). The criterion is $i \in \bigcap_{t=1}^{n_k} \widehat{\mathcal{M}}^{(t)}(\ell_k)$.

Proof of Theorem 8.10. Write M_j for the maximum number of nodes of ℓ_j covered by a node-disjoint set of stems, and M'_j for the same quantity in the graph with a driver added at $i \in \ell_k$. That addition creates no edge, so in the augmented graph every edge still runs from one layer to the next and every stem (rooted in ℓ_1 or at i) still meets each layer in at most one node, over a consecutive block of layers. One step of the proof of Lemma A.2 needs care, since its downward induction is phrased for stems rooted in ℓ_1: above ℓ_k the stem rooted at i contributes nothing, so that induction runs on the ℓ_1-rooted stems alone, and the i-rooted stem rides along from ℓ_k downwards, adding one to the count in every layer from ℓ_k on and nothing

above it. With that reading the largest number of state nodes covered in the augmented graph is $\sum_j M'_j$ and in the original graph $\sum_j M_j$. Every set of the original graph survives in the augmented one, so $M'_j \geq M_j$ in every layer, and the cover count is unchanged exactly when $M'_j = M_j$ throughout. By Theorems 6.4 and 8.7, that is exactly when i is an FSC node.

For the *if* condition, suppose i is matched in every feasible maximum disjoint-stem set for ℓ_k, and fix a layer ℓ_j. A stem rooted at i lies in $\ell_k, \ell_{k+1}, \ldots$, so for $j < k$ nothing changes and $M'_j = M_j$. For $j \geq k$ take a set $\mathcal{S}'$ of the augmented graph with $c_j(\mathcal{S}') = M'_j$. If no stem of it is rooted at i it is a set of the original graph and $M'_j \leq M_j$. Otherwise split it into the stems $\mathcal{S}'_0$ rooted in ℓ_1 and the stem P rooted at i, and let $X \subseteq \ell_k$ collect the nodes of ℓ_k that $\mathcal{S}'_0$ uses. Note that $i \notin X$, since P contains it. Truncated at ℓ_k, $\mathcal{S}'_0$ reaches X by $|X|$ disjoint stems, so the exchange step of Lemma A.2 places X inside the matched set of some maximum disjoint-stem set for ℓ_k, and by hypothesis that matched set contains i as well. Keep the stems ending at $X \cup \{i\}$, re-attach below each node of X what continued below it in $\mathcal{S}'_0$ and below i the stem P. This is legitimate for the reason recorded in Lemma A.2, that the part below ℓ_k depends only on which nodes of ℓ_k are used. Since every stem rooted in ℓ_1 that reaches ℓ_j passes through ℓ_k, the result covers every node of ℓ_j that $\mathcal{S}'$ covered, and it is a set of the original graph: $M_j \geq M'_j$. Hence $M'_j = M_j$ in every layer, and i is an FSC node.

For the *only if* condition, we prove the contrapositive: suppose some feasible maximum disjoint-stem set for ℓ_k fails to match i, and let X be its matched set, so $|X| = M_k$ and $i \notin X$. None of its stems passes through i (one that did would match i), so in the augmented graph those stems and the length-zero stem $\{i\}$ are together node-disjoint and cover $M_k + 1$ nodes of ℓ_k, giving $M'_k \geq M_k + 1$. With $M'_j \geq M_j$ in the remaining layers the number of covered nodes rises, $\dim \mathcal{C}_\Lambda$ increases (Theorem 6.4), and by Theorem 8.7 i is not an FSC node. □

The single-driver corollary (Corollary 8.11) is the case $|\ell_k| = 1$. With one driver ℓ_1 holds a single node, so no two stems are node-disjoint and $M_k = 1$ in every layer. Every node of ℓ_k carries a stem that reaches it at its k-th step, so the feasible maximum disjoint-stem sets for ℓ_k are exactly the stems reaching ℓ_k, one matched node apiece, and their matched nodes run over all of ℓ_k. Node i is matched in every one of them precisely when $|\ell_k| = 1$, which is Theorem 8.10 read in this case. The hierarchy is inherited from that theorem and is needed there: with a skipping edge a node alone in its layer still lies on every longest stem, yet a driver placed on it can free the old stem to take a different path and cover more (Remark 8.12). Layer by layer the criterion is a disjoint-paths computation: joining a super-source to the drivers and a super-sink to ℓ_k turns it into the r-vertex-disjoint-paths problem, where r is the number of drivers: solvable in $O(|\mathcal{E}| + n)$ for a single driver and $O(|\mathcal{E}|\, n^{r-1})$ for $r \geq 2$ drivers on a DAG [30].

Source and notation. Adapted from the author's own [30] (its Theorem 3, the general multi-driver form). The single-driver special case is also in [1]. The source states both results for a general p-layered DAG, in which an edge may skip a layer. The hierarchy is added here because without it the criterion is false (Remark 8.12) and the layerwise decomposition fails (Remark A.3). Notation translated as: the source's "leader" is the book's driver node (occupying ℓ_1), and the source's layers $\mathcal{V}_{l_k}$ are the book's ℓ_k. The source's maximum disjoint stems $\mathcal{M}(\mathcal{V}_{l_k})$ and matched set $\widehat{\mathcal{M}}(\mathcal{V}_{l_k})$ are the book's maximum disjoint-stem set for ℓ_k and its matched set, and the source's "fixed node" is the book's FSC node. The super-source and super-sink of the disjoint-paths reduction are the source's "two dummy nodes s and t". The proof invokes the source's cited Commault theorem, which is the book's Theorem 8.7, and

Hosoe's theorem, the book's Theorem 6.4.

Appendix B

Proofs: Strong Structural Side

The non-trivial proofs behind the SSC column of the book's grid, collected here so the main text can move without interruption. As in the companion Appendix A, each section proves its result in the book's notation and closes with a source-and-notation note recording the origin and the notation translation performed. Most sections restate a result of Part II by cross-reference, and two instead state in full what the main text records only in prose: the shortest-stem lower bound (Section B.1) and the membership criteria for the two fixed subspaces (Section B.7). The SSC side asks for controllability for every realization, so the arguments below rest on Lemma 7.8 ("full row rank for all realizations" as a dedicated-node count), proved once in Chapter 7 and reused below without reproof.

B.1 Shortest-Stem Lower Bound on $\dim \mathcal{C}_\Gamma$

Section 6.4 records, in prose, that the worst-case dimension is bounded below by the number of state nodes on a longest shortest stem. We promote that remark to a theorem and prove it. Recall (Definition 4.9) that a *stem* is a path from an input node. Call a stem $\sigma = (u, v_1, v_2, \ldots, v_q)$ (input u followed by q state nodes) a *shortest stem* if for every j the node v_j sits at input-distance j (Definition 4.8: the distance measured from the input nodes as a whole) and $(u, v_1, \ldots, v_j)$ is the only walk of that length from an input node to v_j, so that the corresponding entry of $\mathcal{C}_F$ is a single-term. (With one input this is exactly the standing hypothesis of Section 6.4: on the diamond the stem $u{\to}1{\to}2$ qualifies, its entries 1 and a_{21} being single-terms. With several inputs it matters that the distance is taken over all of them. See Remark B.2.) Let $q^\star$ be the largest number of state nodes on any such stem. In this appendix p counts the inputs (the m of Eq. (2.1)), re-lettered because the letter m is taken here by $m^\star$ and, in Section B.5, by the sink count.

Theorem B.1 (Shortest-stem lower bound)**.** *For a single-input pattern,* $\dim \mathcal{C}_\Gamma \geq q^\star$. *More generally, with p inputs, let $m^\star$ be the largest number of state nodes that can be covered by p node-disjoint shortest stems, and suppose that for every node l and every step count k there is at most one k-step stem to l from* any *input node (multiple stems to one node are allowed, provided their step counts differ). Then* $\dim \mathcal{C}_\Gamma \geq m^\star$.

Proof. Single input. Let the input be u with input column b, so $\mathcal{C}_F = [\,b,\ A_F b,\ \ldots,\ A_F^{n-1} b\,]$, and fix a shortest stem $\sigma = (u, v_1, \ldots, v_q)$ with $q = q^\star$ nodes. By Proposition 4.24 the entry of

$A_F^{j-1}b$ at row v_j is the SWP of all length-$(j-1)$ walks from the driver node v_1 to v_j. Because σ is a shortest stem, node v_j sits at input-distance j and its shortest walk is unique, so this entry is the single non-zero term $\prod_{i=1}^{j-1}[A_F]_{v_{i+1}v_i} \neq 0$.

Permute rows and columns so that rows $v_1, \ldots, v_q$ and columns $b, A_F b, \ldots, A_F^{q-1}b$ come first, and let M be the resulting $q \times q$ leading submatrix. Its $(v_w, \text{column } j)$ entry is the row-v_w coefficient of $A_F^{j-1}b$. For $w > j$ this entry would require a walk from the driver node v_1 to v_w of length $j-1 < w-1$, contradicting that v_w lies at distance w on the shortest stem. Hence all such entries vanish and M is triangular. Its diagonal entries (at $w = j$) are the single non-zero terms above, so

$$\det M \;=\; \prod_{j=1}^{q}\prod_{i=1}^{j-1}[A_F]_{v_{i+1}v_i} \;\neq\; 0 \qquad \text{for every realization.}$$

Thus $\operatorname{rank}\mathcal{C}_F \geq q$ for *every* $A_F \in \mathcal{Q}(A)$, and taking the worst case, $\dim\mathcal{C}_\Gamma \geq q = q^\star$.

Multiple inputs. Let $\sigma_1, \ldots, \sigma_p$ be p node-disjoint shortest stems covering $m^\star$ state nodes, stem σ_r rooted at input u_r with q_r state nodes, input column b_r, and j-th state node v_j^r. Select the $m^\star = \sum_r q_r$ rows v_j^r (distinct, by disjointness) and the $m^\star$ columns $A_F^{j-1}b_r$ with $1 \leq j \leq q_r$, written (r, j). Order rows and columns by the step count j, *not* by stem. By Proposition 4.24 the entry of column (r, j) at row $v_w^{r'}$ collects the length-$(j-1)$ walks from v_1^r to $v_w^{r'}$, that is, the j-step stems from u_r to that node. Two counts settle the submatrix.

For $j < w$ the entry vanishes: $v_w^{r'}$ sits at input-distance w, so no walk from any input node reaches it in j steps. For $j = w$ every such walk has the minimum length, hence is a w-step stem to $v_w^{r'}$. The definition of a shortest stem (and equally the step-count hypothesis) leaves exactly one of those, the prefix of $\sigma_{r'}$, rooted at $u_{r'}$. That entry is therefore zero unless $r = r'$, when it is the prefix's single non-zero weight product. So the submatrix is triangular by step count, each step-count group carrying one non-zero entry per stem, and its determinant is, up to sign, the product over r of the single-input monomial displayed above for σ_r, which is non-zero for every realization. Thus $\operatorname{rank}\mathcal{C}_F \geq m^\star$ for every $A_F \in \mathcal{Q}(A)$ and $\dim\mathcal{C}_\Gamma \geq m^\star$. (Entries with $j > w$ are unconstrained. They sit above the diagonal and do not enter the determinant.) □

The bound is tight on the diamond: the shortest stem $u{\to}1{\to}2$ gives $q^\star = 2$, and the cancellation of Example 6.2 shows $\dim\mathcal{C}_\Gamma = 2$ exactly.

Remark B.2 (Why both hypotheses are needed). Take inputs at $1, 2$ and the edges $1{\to}3$, $1{\to}5$, $2{\to}4$, $4{\to}3$, $4{\to}5$. No node is reached by two stems of the same step count, so the step-count hypothesis holds. Measuring each stem against its own input alone would admit $u_1{\to}1{\to}3$ and $u_2{\to}2{\to}4{\to}5$, two disjoint stems covering all five nodes. Yet, discarding the zero columns,

$$\mathcal{C}_F = [\, e_1,\; e_2,\; a_{31}e_3 + a_{51}e_5,\; a_{42}e_4,\; a_{42}(a_{34}e_3 + a_{54}e_5)\,]$$

has rank 4 whenever $a_{31}a_{54} = a_{51}a_{34}$, so $\dim\mathcal{C}_\Gamma = 4$: node 5 sits at input-distance 2, reached by $u_1{\to}1{\to}5$, while the second stem needs three steps to reach it, and the two stems' single-terms then cancel against each other in the minor on nodes 3 and 5. Under the input-distance of Definition 4.8 that stem stops at node 4, giving $m^\star = 4$, which the bound attains. The uniqueness that the step-count hypothesis secures is equally indispensable: on $1{\to}3$, $1{\to}4$, $2{\to}3$, $2{\to}4$ with inputs at $1, 2$, a count that ignored it would again give $m^\star = 4$, yet

$$\mathcal{C}_F = [\, e_1,\; e_2,\; a_{31}e_3 + a_{41}e_4,\; a_{32}e_3 + a_{42}e_4\,]$$

has rank 3 whenever $a_{31}a_{42} = a_{32}a_{41}$, node 3 carrying two two-step stems, one from each input node.

Source and notation. Adapted from the author's dissertation [1] (its single- and multiple-input shortest-stem theorems) with path-enumeration support from [12, 4]. Notation translated as: the source writes A_Λ for the generic realization inside this bound's proof (inessential there, since the argument holds for *every* realization), which is the book's A_F. The source's $|\mathcal{C}_\Gamma|$ is $\dim \mathcal{C}_\Gamma$ (Γ the worst case, as in the book). The row-swap to a triangular submatrix in the single-input proof is the source's, rewritten in the book's notation.

Two departures from the source are recorded here, both needed for the multi-input clause to be true. First, the source's standing Assumption is not a per-stem uniqueness but the statement the theorem now carries: for every node l and every step count k there is at most one k-step stem to l from *any* input node, several stems to one node being allowed when their step counts differ. The proof draws on it only at the input-distance, where the definition of a shortest stem already secures the uniqueness. The theorem states it in full because that is the form the source assumes and the form Section 6.4 quotes. Second, the source measures a shortest stem against its own input node, which for a single input is the input-distance of Definition 4.8 but for several is weaker. The definition above takes the distance over all input nodes. Without that, a stem may run past a node that a different input node reaches in fewer steps, and the two single-terms so produced cancel for some realization. The source's multi-input proof concludes instead, from disjoint full-rank diagonal blocks, that the ranks add. That does not follow for a submatrix with arbitrary off-diagonal entries, so the proof above replaces the step by the ordering on step count, under which the whole submatrix is triangular.

B.2 Mayeda's Theorem: Two Dedicated-Node Conditions

Theorem 7.9 characterizes SSC of a single-attachment pattern by two conditions on subsets $S \subseteq \mathcal{V}^S$: (C1) every non-empty S has a dedicated node in $N^{\text{in}}(S)$ (possibly inside S), and (C2) every non-empty S with $S \subseteq N^{\text{in}}(S)$ has a dedicated node in $N^{\text{in}}(S) \setminus S$. We prove the equivalence with SSC in book notation, building on Lemma 7.8 (full row rank for all realizations, proved in Chapter 7, not reproved here) and the PBH test Theorem 3.8. Throughout, recall the edge convention (Definition 2.3): column v of $[A_F\ B]$ has its non-zero rows exactly at the nodes v drives, so "a column with exactly one non-zero in the rows S" is precisely a dedicated node of S (Definition 7.7). Where the pattern carries a self-loop at a node of S, that loop is one of its edges into S.

Proof of Theorem 7.9. The family is SSC iff every realization is controllable, iff (PBH, Theorem 3.8) for every realization and every $\lambda \in \mathbb{C}$ the matrix $[\,A_F - \lambda I \;\; B\,]$ has full row rank n. We split on λ.

The $\lambda = 0$ regime is (C1). At $\lambda = 0$ the test matrix is $[\,A_F \;\; B\,]$. By Lemma 7.8 its pattern has full row rank at every realization iff every non-empty row set $S \subseteq \mathcal{V}^S$ admits a column with exactly one non-zero in S, i.e. a dedicated node of S. A member $j \in S$ can serve, through its single edge into S. The dedicated node ranges over all nodes, state or input, exactly as (C1) states. (B is held fixed and binary (Section 4.1), so $[\,A_F \;\; B\,]$ is not literally a pattern matrix. Single attachment repairs this, since a column of B carrying exactly one non-zero never has

two entries inside a row set and so is never retuned by the construction of Lemma 7.8.) Hence full row rank for all realizations at $\lambda = 0 \Leftrightarrow$ (C1).

The $\lambda \neq 0$ regime is (C2). Fix $\lambda \neq 0$. We show $[\,A_F - \lambda I \;\; B\,]$ has full row rank for all realizations iff (C2) holds.

For the *if* condition, assume (C2). For contradiction, suppose that some realization admits $z \neq 0$ with $z^\top [\,A_F - \lambda I \;\; B\,] = 0$. Let $S = \mathrm{supp}(z) \neq \emptyset$. If some node $s \in S$ has no out-neighbor inside S, it has in particular no self-loop, so $a_{ss} = 0$ and column s of $A_F - \lambda I$ meets the rows S only in its diagonal entry $[A_F - \lambda I]_{ss} = -\lambda$. Then $0 = z^\top (A_F - \lambda I) e_s = -\lambda\, z_s$ forces $z_s = 0$, contradicting $s \in \mathrm{supp}(z)$. Hence every node of S has an out-neighbor inside S, and (C2) supplies a dedicated node v outside S with a single edge into S, say $v \to j_0$ with $j_0 \in S$. If v is a state node, column v meets the rows S only at row j_0 (value $[A_F]_{j_0 v} \neq 0$, and $v \notin S$ contributes no diagonal term), so $0 = z^\top (A_F - \lambda I) e_v = z_{j_0} [A_F]_{j_0 v}$ gives $z_{j_0} = 0$. If v is an input, the same computation on its column of B, using $z^\top B = 0$, gives $z_{j_0} = 0$. Either way $j_0 \in S$ is contradicted. So no such z exists and the row rank is full.

For the *only if* condition, we prove the contrapositive: suppose that (C2) fails, so that some non-empty S in which every node has an out-neighbor inside S has *no* dedicated node outside S, that is, every node outside S with an edge into S has at least two. Build a realization and a null vector as follows. Set $\lambda = 1$ and $z = \mathbf{1}_S$ (the indicator of S). Column by column of $[\,A_F - I \;\; B\,]$:

- for $j \in S$: since j has an out-neighbor inside S, the free weights $[A_F]_{ij}$ on its out-edges into S can be chosen to sum to $1 = z_j$, making $z^\top (A_F - I) e_j = \sum_{i \in S} [A_F]_{ij} - z_j = 0$. These entries belong to column j alone, so the choices never conflict.
- for a state node $j \notin S$: if j has edges into S it has at least two free weights $[A_F]_{ij}$, chosen to sum to 0, and if it has none the sum is empty. Either way $z^\top (A_F - I) e_j = 0$.
- for an input column: a single-attachment input driving into S would be a dedicated node outside S, excluded by the failure of (C2), so every input drives a node outside S and $z^\top B = 0$ automatically.

This realization has $z^\top [\,A_F - I \;\; B\,] = 0$ with $z \neq 0$, so PBH fails at $\lambda = 1$ and the family is not SSC. Likewise if (C1) fails, some S has every column of $[A_F \; B]$ carrying none or ≥ 2 non-zeros in S, and Lemma 7.8 produces a realization with dependent rows at $\lambda = 0$: not SSC. Contrapositively, SSC forces both (C1) and (C2). □

The two regimes are genuinely independent: (C1) is the $\lambda = 0$ test and (C2) the $\lambda \neq 0$ test, and the four-node network passes both (Example 7.10) while the diamond and the twin star already fail (C1).

Source and notation. The result is Mayeda's [17] in the modern algebraic form of [24], with the dedicated/sharing-node reading of [1, 23] and the survey [4]. Notation translated as: the source's α is the book's S, and the neighboring set $\mathcal{N}(\alpha)$ of the sources is the book's $N^{\mathrm{in}}(S)$ (Definition 4.5), the direction the λ-split proof requires. The substantive translation, flagged in the Chapter 7 scope comment, is that the dedicated node may be an *input* node. Without that both conditions fail on the four-node pattern. In (C1) a member of S may itself serve as the dedicated node, which the $\lambda = 0$ application of Lemma 7.8 yields directly. The source's $\mathcal{C}_\Gamma$ is the book's $\mathcal{C}_\Gamma$, and its input-attachment convention is the single-attachment standing convention of Section 4.1. The dissertation's own single-condition form (Lemma 7.8's $\lambda = 0$ test alone) is the *self-loop* regime treated next, not general Mayeda.

B.3 Zero-Forcing Criterion under Free Self-Loops

Theorem 7.14 states that when every diagonal entry of A_F is a free parameter (a self-loop of arbitrary weight, zero included, at every state node), the family is controllable for all realizations iff the driver set is a zero forcing set (Definition 7.13), decidable in polynomial time. We prove it by collapsing the two Mayeda conditions of Section B.2 to a single one and identifying that one with the color-change rule.

Proof of Theorem 7.14. Collapse to one condition. With a free diagonal, the shift $a_{ii} \mapsto a_{ii} - \lambda$ maps the free parameter onto itself, so ranging over all realizations already realizes every real shift (a non-real λ only makes $a_{ii} - \lambda$ harder to cancel, so it imposes nothing new). Running the PBH test of Section B.2 with a_{ii} free absorbs λ into the diagonal. Concretely, re-examine the two regimes. Every state node j now carries a self-loop, hence an out-edge into any S it belongs to, so:

- a loop entry is free *including zero*, so it is never a guaranteed non-zero: only a structurally non-zero pattern edge, or an input column, can supply the single non-zero that Lemma 7.8 demands. A member $j \in S$ whose only entry in the rows S is its own loop therefore supplies no such non-zero.
- a member $j \in S$ that does feed another member has a column carrying that edge alongside the loop entry, which is free to be non-zero, so at some realization it has two non-zeros in S.

Either way no member of S can serve as the guaranteed single non-zero that Lemma 7.8 demands, so the dedicated node must come from *outside* S.

By the argument of Lemma 7.8, adapted to a diagonal that is free *including zero* (such an entry can be set to zero, so it never supplies the required single non-zero, and can be set non-zero, so it never blocks a cancellation), full row rank of $[\, A_F \;\; B \,]$ for all realizations, a test that now subsumes the $\lambda \neq 0$ regime, is therefore equivalent to the single condition

$$\begin{aligned} &(\mathrm{C}^\star)\text{: every non-empty } S \subseteq \mathcal{V}^S \text{ has a dedicated node} \\ &\quad \text{in } N^{\mathrm{in}}(S) \setminus S. \end{aligned} \tag{B.1}$$

Both loop-free cases of Theorem 7.9 have disappeared: an inside dedicated node, and a node with no out-neighbor inside S, which (C2) does not constrain. This is the sense of the main text's compact phrase "every S has a dedicated node outside S".

Forcing decides (B.1). A black node with exactly one white out-neighbor is a node with exactly one edge into the current white set (a dedicated node, lying outside that set), so a forcing step exhibits exactly the outside dedicated node required by (B.1). Running the color-change rule from the drivers until it can no longer be applied computes the derived set. The driver set is a zero forcing set iff the derived set is all of $\mathcal{V}^S$, iff no non-empty white set survives without an outside dedicated node, iff (B.1) holds. This equivalence, and that the rule reaches that point in $O(n\,|\mathcal{E}|)$ time, are the content of [25, 28]. Combining, the free-diagonal family is SSC iff the driver set is a zero forcing set, in polynomial time. □

The self-loop hypothesis cannot be dropped. Without it the equivalence breaks in one direction only: forcing remains *sufficient* for SSC of a loop-free pattern (the free-diagonal family contains the zero-diagonal one), but not necessary, the four-node network of Example 7.15 being SSC while the process stops with a derived set smaller than $\mathcal{V}^S$. Its remaining white set

$\{3, 4\}$ has an *inside* dedicated node and a node with no out-neighbor inside it, the very cases that (B.1) discards. For loop-free patterns the exact criterion remains Theorem 7.9.

Source and notation. The equivalence is Monshizadeh, Zhang and Camlibel [25], with the dedicated-node collapse and the loop-free restriction from the author's [28] and the survey [4]. Notation translated as: the source's qualitative class with free self-loops is the free-diagonal family here, and its "zero forcing set" and "color-change rule" are used verbatim (Definition 7.13). The collapse is expressed through Lemma 7.8 and the Mayeda proof of Section B.2 rather than re-derived, and the four-node counterexample is Example 7.15.

B.4 Fixed-Node Input-Addition Test: Min-Dimension Side

We prove the FSSC half of Theorem 8.7: if node k is an FSSC node ($e_k \in \mathcal{C}_\Gamma^F$, Definitions 8.2 and 8.6), then adding a dedicated input at k, replacing B by $B_k = [\,B,\ e_k\,]$, does not increase $\dim \mathcal{C}_\Gamma$. Only that direction holds, so on the SSC side the dimension comparison is a necessary condition and no more: a strict increase rules k out, a non-increase decides nothing (Remark 8.8). The exact criterion is the subspace equality of Lemma B.3 below, asked at every realization, which is the index set $\mathcal{C}_\Gamma^F$ is intersected over (Definition 8.2). The SC side can make do with a comparison of dimensions because $\dim \mathcal{C}_\Lambda$ is a maximum, so one generic realization missing e_k already lifts it. In contrast, $\dim \mathcal{C}_\Gamma$ is a minimum, and a realization missing e_k need not move it. (The FSC half, with $\dim \mathcal{C}_\Lambda$, is proved in the companion appendix, Section A.4, and stays an equivalence.) Write $\mathcal{C}_{F,k}$ for the controllability matrix of the augmented pair (A_F, B_k) and $\mathcal{C}_{\Gamma,k}$ for its strongly structurally controllable subspace. Since $\mathcal{C}_{F,k}$ contains the columns of $\mathcal{C}_F$, for every realization $\operatorname{rank} \mathcal{C}_{F,k} \geq \operatorname{rank} \mathcal{C}_F$. Taking worst cases, $\dim \mathcal{C}_{\Gamma,k} \geq \dim \mathcal{C}_\Gamma$ always.

The argument, as on the SC side, rests on the A_F-invariance of the controllable subspace: for a fixed realization

$$\operatorname{Im} \mathcal{C}_{F,k} \;=\; \operatorname{Im} \mathcal{C}_F \;+\; \operatorname{span}\{e_k, A_F e_k, \ldots, A_F^{n-1} e_k\}, \tag{B.2}$$

because $\operatorname{Im} \mathcal{C}_{F,k}$ is the smallest A_F-invariant subspace containing both $\operatorname{Im} B$ and e_k (Definition 3.3).

Lemma B.3 (Augmented subspace)**.** *$e_k \in \mathcal{C}_\Gamma^F$ if and only if $\operatorname{Im} \mathcal{C}_{F,k} = \operatorname{Im} \mathcal{C}_F$ at every realization $A_F \in \mathcal{Q}(A)$.*

Proof. For the *only if* condition, let $e_k \in \mathcal{C}_\Gamma^F = \bigcap_{A_F \in \mathcal{Q}(A)} \operatorname{Im} \mathcal{C}_F$ (Definition 8.2) and fix a realization A_F. Then $e_k \in \operatorname{Im} \mathcal{C}_F$, and $\operatorname{Im} \mathcal{C}_F$ is A_F-invariant, so $A_F^j e_k \in \operatorname{Im} \mathcal{C}_F$ for every j: the span in (B.2) is already contained in $\operatorname{Im} \mathcal{C}_F$, whence $\operatorname{Im} \mathcal{C}_{F,k} = \operatorname{Im} \mathcal{C}_F$ at that realization. The realization was arbitrary, so the equality holds throughout the family. For the *if* condition, suppose that $\operatorname{Im} \mathcal{C}_{F,k} = \operatorname{Im} \mathcal{C}_F$ at every realization. Then $e_k \in \operatorname{Im} \mathcal{C}_{F,k} = \operatorname{Im} \mathcal{C}_F$ at each, so e_k lies in their intersection, i.e. $e_k \in \mathcal{C}_\Gamma^F$. □

Proof of Theorem 8.7, FSSC part. Suppose $e_k \in \mathcal{C}_\Gamma^F$. By Lemma B.3, $\operatorname{Im} \mathcal{C}_{F,k} = \operatorname{Im} \mathcal{C}_F$ at every realization, so $\operatorname{rank} \mathcal{C}_{F,k} = \operatorname{rank} \mathcal{C}_F$ throughout the family. The two minima therefore agree, $\dim \mathcal{C}_{\Gamma,k} = \dim \mathcal{C}_\Gamma$: no increase. □

The converse fails, and it is worth seeing where. From $\dim \mathcal{C}_{\Gamma,k} = \dim \mathcal{C}_\Gamma$ one does obtain the subspace equality *somewhere*: at any realization attaining the augmented minimum,

$\operatorname{rank}\mathcal{C}_{F,k} = \dim\mathcal{C}_\Gamma \le \operatorname{rank}\mathcal{C}_F \le \operatorname{rank}\mathcal{C}_{F,k}$ forces $\operatorname{rank}\mathcal{C}_{F,k} = \operatorname{rank}\mathcal{C}_F$, and the two images are nested, so they coincide there. What Lemma B.3 asks for is that equality at *every* realization, and a comparison of two minima cannot reach it: the argument forces the equality only where the augmented minimum is attained, and those may be the only realizations at which it holds. Attach a new node 5 to the diamond by the edge $2 \to 5$ (Remark 8.8). Its $\operatorname{rank}\mathcal{C}_F$ is 3 at every realization, because the e_5 entry $a_{52}a_{21}$ of its third column is a single-term. The augmented $\operatorname{rank}\mathcal{C}_{F,5}$ is 4 except on $a_{42}a_{21} + a_{43}a_{31} = 0$, where it is again 3, so the augmented minimum is still 3 and the dimension test reports no increase. Yet off that locus the third column $(a_{42}a_{21} + a_{43}a_{31})e_4 + a_{52}a_{21}e_5$ mixes e_4 with e_5 and $e_5 \notin \operatorname{Im}\mathcal{C}_F$, so $e_5 \notin \mathcal{C}_\Gamma^F$: the equality holds on the locus and nowhere else. The exact criterion is therefore Lemma B.3, a subspace equality at every realization. The dimension comparison of Theorem 8.7 is on this side only a necessary test, useful for ruling nodes out, as is, by Proposition B.5, the SC-side comparison, since every FSSC node is an FSC node.

Source and notation. Author's own result [12] (its augmented-subspace lemma and FSSC-node theorem). Notation translated as: the source's $\mathcal{C}_\Gamma$ is the book's $\mathcal{C}_\Gamma$, its $\mathcal{C}_\Gamma^{\mathcal{F}}$ the book's $\mathcal{C}_\Gamma^F$, its standard basis v_i the book's e_k, and its augmented input $B_{v_i} = [B, v_i]$ the book's $B_k = [\,B,\ e_k\,]$. "i becoming a leader with additional input" is the dedicated-input addition. One departure is recorded here. What the source establishes, and what Lemma B.3 keeps word for word, is the *subspace* equality $\mathcal{C}_{\Gamma_{v_i}} = \mathcal{C}_\Gamma$ "for all network parameters", that is, at every realization, which is the quantifier the lemma carries and the one under which $\mathcal{C}_\Gamma^F$ is read throughout the book (Section B.7). The source then restates that equality as a comparison of the dimensions of the two subspaces, and the restatement does not survive, not even on the hierarchical DAGs (Definition 4.21) the source works over. The diamond with the added node 5 above is one. The book therefore keeps only the necessary direction in Theorem 8.7 and leaves the criterion with Lemma B.3. The computability caveat (the test is effective exactly where $\dim\mathcal{C}_\Gamma$ is, on hierarchical DAGs) is main text prose (Section 8.4), not part of the proof.

B.5 Shortest-Path Criterion for SSC Nodes

Theorem 8.17 localizes the dedicated-node conditions to a single node on a polytree: for an input-connected polytree with a single input node u and sinks $t_1, \dots, t_m$, node k is an SSC node if and only if k belongs to the intersection $\mathcal{V}^u_{\text{int}}$ of the shortest-path graphs (control paths) from u to $t_1, \dots, t_m$. Recall Definition 8.16: a set $S \subseteq \mathcal{V}^S$ is a *controllable subset* if some node outside S sends exactly one edge into S, and k is an *SSC node* if every $S \subseteq \mathcal{V}^S$ containing k is a controllable subset.

One structural fact underlies the proof. An input-connected polytree with a single input u is a tree rooted at u with every edge oriented away from u: the undirected tree has a unique path from u to each node, and reachability from u fixes that orientation, so every state node has exactly one in-edge, from its unique in-neighbor. Membership $k \in \mathcal{V}^u_{\text{int}}$ then says that k lies on the path from u to every sink. Equivalently, the path $u = w_0 \to w_1 \to \cdots \to w_s = k$ is *unbranched* (each w_i with $i < s$ has w_{i+1} as its only out-neighbor), since any branch off that path would carry a sink not reachable from k.

Proof of Theorem 8.17. For the *if* condition, let $k \in \mathcal{V}^u_{\text{int}}$, so the path $w_0 \to \cdots \to w_s = k$ is unbranched, and take any $S \subseteq \mathcal{V}^S$ with $k \in S$. Since $k = w_s \in S$, some node of that path lies in S. Let w_j be the one of least index. Since S contains no input node, $j \ge 1$. Its in-neighbor

w_{j-1}, the input u itself when $j = 1$, lies outside S (by minimality when $j \geq 2$), and being on the unbranched path it has w_j as its *only* out-neighbor, hence exactly one edge into S. So w_{j-1} is a dedicated node of S outside S: S is a controllable subset. As S was arbitrary, k is an SSC node.

For the *only if* condition, we prove the contrapositive: let $k \notin \mathcal{V}^u_{\text{int}}$ and set $S = \mathcal{V}^S \setminus \mathcal{V}^u_{\text{int}}$, which contains k. Let v be the last node of the path formed by $\mathcal{V}^u_{\text{int}}$, the node at which the path branches. Every node strictly above v on that path has its single out-neighbor inside $\mathcal{V}^u_{\text{int}}$, hence no edge into S. The input u attaches inside $\mathcal{V}^u_{\text{int}}$ as well. So the only node *outside* S with edges into S is v itself, and beyond v the paths to different sinks diverge, so v has at least two out-neighbors, each opening a branch that lies in S. Thus v sends at least two edges into S and is a sharing node, leaving S with no dedicated node outside S: S is not a controllable subset. Since $k \in S$, k is not an SSC node. □

For several inputs, taking the union over inputs of the per-input intersections $\mathcal{V}^{u_j}_{\text{int}}$ still yields SSC nodes:

Corollary B.4 (Multiple inputs)**.** *On an input-connected polytree with inputs $u_1, \ldots, u_p$, every node in $\bigcup_j \mathcal{V}^{u_j}_{\text{int}}$ is an SSC node.*

Proof. If $k \in \mathcal{V}^{u_j}_{\text{int}}$ for some input u_j, the unbranched-path argument of the *if* condition, applied within the part reachable from u_j, furnishes a dedicated node outside each $S \ni k$ along the u_j-path. The presence of further inputs only adds more candidate dedicated nodes. Hence every such S is a controllable subset and k is an SSC node. □

Source and notation. Author's own [31] (its single-input path theorem and multi-input corollary), with the controllable-subset and SSC-node definitions of [23]. Notation translated as: the source's "acyclic once the edge directions are ignored" is the book's *polytree*, and its *terminal nodes* are the book's sinks (Definition 4.20). The source's subsets α are written S here, as in Theorem 7.9. Its shortest-path graphs $\mathcal{P}^u_i$ and their intersection $\mathcal{P}^u_{\text{int}}$ are the book's control paths and their intersection $\mathcal{V}^u_{\text{int}}$, and the union graph $\mathcal{P}^c_{\text{int}}$ over inputs is $\bigcup_j \mathcal{V}^{u_j}_{\text{int}}$. "SSC node" is used in the book's sense (every subset containing k controllable), which is the dissertation's, *not* the survey's min-dimension usage (Remark 8.20). The rooted-tree observation makes the source's "$\mathcal{P}^u_{\text{int}}$ is a path" step explicit in the book's notation.

B.6 SSC Nodes Are FSC Nodes

Proposition 8.18 states that on an input-connected polytree with a single input node every SSC node is an FSC node, and that with several inputs the same conclusion holds for every node of the union $\bigcup_j \mathcal{V}^{u_j}_{\text{int}}$ of the per-input intersections. The main text gives only a one-line sketch. Here is the argument.

Proof of Proposition 8.18. Single input. Let k be an SSC node of an input-connected polytree with input u. By Theorem 8.17, $k \in \mathcal{V}^u_{\text{int}}$: the path $u \to \cdots \to k$ is unbranched, so a directed path from u either stops above k or passes through it. In particular, every longest stem contains k. With a single input the maximum disjoint stem-and-cycle cover of Theorem 6.4 is realized by a single stem (a longest directed path from u), and every such stem therefore contains k. Adding a dedicated input at k changes which stem covers what, but not how much: the path from u down to k is unbranched, so a u-rooted stem that avoids k stops above it and

covers only the nodes strictly between u and k, while the stem rooted at k covers k and a longest path below it. Together they cover exactly the nodes of a longest path from u, the original count. The cover count, hence $\dim \mathcal{C}_\Lambda$, is unchanged. By the FSC half of Theorem 8.7 (proved in Section A.4), k is an FSC node.

Multiple inputs. Here the hypothesis is the one the several-input clause carries: $k \in \mathcal{V}_{\text{int}}^{u_j}$ for some input u_j, which by Corollary B.4 already makes k an SSC node. Within the part reachable from u_j the path $u_j \to \cdots \to k$ is unbranched and every u_j-rooted stem passes through k. In a maximum disjoint stem-and-cycle cover the stem rooted at u_j then contains k, so a dedicated input at k re-traces that stem and raises no cover count. Thus $\dim \mathcal{C}_\Lambda$ is unchanged and k is an FSC node by the FSC half of Theorem 8.7. □

Note what the antecedent is here: an *SSC node* (the subset/path notion), not the FSSC-node membership $e_k \in \mathcal{C}_\Gamma^F$, and neither reading is claimed to imply the other (Remark 8.20). The pruned diamond of Example 8.19 has node 4 FSSC but not an SSC node. The FSSC nodes reach the same conclusion by a different and much shorter argument: they are FSC nodes because $\mathcal{C}_\Gamma^F \subseteq \mathcal{C}_\Lambda^F$ (Proposition B.5), with no polytree hypothesis and no appeal to Theorem 8.17.

Source and notation. Author's own [31]: its single-input theorem, which reads "every SSC node is a fixed controllable node", and its multi-input theorem, which reads "every node of $\bigcup_j \mathcal{V}_{\text{int}}^{u_j}$ is a fixed controllable node". Notation translated as before: SSC node in the book/dissertation sense, control paths $\mathcal{V}_{\text{int}}^{u_j}$, and the FSC input-addition test = the source's cited Commault criterion = book Theorem 8.7. Each half proceeds through Theorem 8.17 exactly as the source does through its path theorem. With one input the two hypotheses coincide there, so that half covers every SSC node.

B.7 Inclusion and Membership for the Two Fixed Subspaces

Section 8.2 leaves the relation between the two fixed subspaces to the appendix. Both are intersections of $\operatorname{Im} \mathcal{C}_F$, but over different sets of realizations: $\mathcal{C}_\Lambda^F$ over the generic ones, those attaining $\dim \mathcal{C}_\Lambda$, and $\mathcal{C}_\Gamma^F$ over the whole family (Definition 8.2). A direction of $\mathcal{C}_\Gamma^F$ is controllable at *every* realization, which is what an FSSC node asks for. One index set contains the other, so an inclusion follows at once, and the two input-addition sections already carry the membership test that goes with each side.

Proposition B.5 (Inclusion and membership)**.** *Let $v \in \mathbb{R}^n$, let $B_v = [\,B,\ v\,]$ be the input matrix augmented by v, and let $\mathcal{C}_{F,v}$ be the controllability matrix of the pair (A_F, B_v).*

1. *$\mathcal{C}_\Gamma^F \subseteq \mathcal{C}_\Lambda^F$, and in particular every FSSC node is an FSC node.*
2. *$v \in \mathcal{C}_\Gamma^F$ if and only if $\operatorname{Im} \mathcal{C}_{F,v} = \operatorname{Im} \mathcal{C}_F$ at every realization.*
3. *$v \in \mathcal{C}_\Lambda^F$ if and only if $\operatorname{Im} \mathcal{C}_{F,v} = \operatorname{Im} \mathcal{C}_F$ at every generic realization, and equivalently if and only if augmenting B by v leaves $\dim \mathcal{C}_\Lambda$ unchanged.*
4. *Every driver axis lies in $\mathcal{C}_\Gamma^F$, hence in both subspaces, and if $\dim \mathcal{C}_\Gamma = \dim \mathcal{C}_\Lambda$ then $\mathcal{C}_\Gamma^F = \mathcal{C}_\Lambda^F$.*

Proof. (i) The generic realizations form a non-empty subset of $\mathcal{Q}(A)$, non-empty because $\dim \mathcal{C}_\Lambda$ is attained (Lemma A.1). Intersecting a collection of sets over a larger index set can

only make the intersection smaller:

$$\mathcal{C}_\Gamma^F = \bigcap_{A_F \in \mathcal{Q}(A)} \operatorname{Im} \mathcal{C}_F \subseteq \bigcap_{\substack{A_F \in \mathcal{Q}(A) \\ \operatorname{rank} \mathcal{C}_F = \dim \mathcal{C}_\Lambda}} \operatorname{Im} \mathcal{C}_F = \mathcal{C}_\Lambda^F.$$

Applying this to $v = e_k$ gives the node-level form.

(ii) The identity (B.2) does not use that the added column is a standard basis vector: for any fixed v, $\operatorname{Im} \mathcal{C}_{F,v} = \operatorname{Im} \mathcal{C}_F + \operatorname{span}\{v, A_F v, \ldots, A_F^{n-1} v\}$, because $\operatorname{Im} \mathcal{C}_{F,v}$ is the smallest A_F-invariant subspace containing both $\operatorname{Im} B$ and v (Definition 3.3). Both directions of Lemma B.3 then read verbatim with v in place of e_k: if v lies in every $\operatorname{Im} \mathcal{C}_F$, invariance closes the added span inside $\operatorname{Im} \mathcal{C}_F$ at each realization. Conversely, the equality puts v in every $\operatorname{Im} \mathcal{C}_F$.

(iii) The same two steps run over the generic realizations alone and give the first equivalence. The second is the proof of the FSC half of Theorem 8.7 in Section A.4, whose only use of e_k is through (A.1).

(iv) A driver axis e_k is a column of B, so it lies in $\operatorname{Im} \mathcal{C}_F$ at every realization and hence in $\mathcal{C}_\Gamma^F$. By (i) it lies in $\mathcal{C}_\Lambda^F$ too. If $\dim \mathcal{C}_\Gamma = \dim \mathcal{C}_\Lambda$ then $\operatorname{rank} \mathcal{C}_F$, which lies between that minimum and that maximum, takes the one value at every realization. Every realization is then generic, the index set of $\mathcal{C}_\Lambda^F$ is the whole family, and the two intersections are the same subspace. □

Clause (i) turns the SC-side machinery into a necessary condition for the SSC side: if adding a dedicated input at k raises $\dim \mathcal{C}_\Lambda$, then k is not an FSC node (Theorem 8.7) and so not an FSSC node either, a polynomial test (Theorem 6.4) that rules FSSC nodes out without computing $\dim \mathcal{C}_\Gamma$ at all. It is only a necessary condition: the diamond has FSC nodes $\{1, 4\}$ and FSSC node 1 alone (Example 8.9), so passing it decides nothing.

What clause (i) does depend on is that the $\mathcal{C}_\Gamma^F$ intersection runs over the whole family. Restricting it to the realizations attaining $\dim \mathcal{C}_\Gamma$ names a different subspace, and that subspace need not lie inside $\mathcal{C}_\Lambda^F$. The pattern of Remark 8.5 (driver 1 with state edges $1 \to 2$, $1 \to 3$, $2 \to 4$, $2 \to 5$, $3 \to 5$, $5 \to 6$) shows it. Its non-zero columns are e_1, then $a_{21}e_2 + a_{31}e_3$, then $a_{21}a_{42}e_4 + (a_{21}a_{52} + a_{31}a_{53})e_5$, then $(a_{21}a_{52} + a_{31}a_{53})a_{65}e_6$, so $\dim \mathcal{C}_\Lambda = 4$, $\dim \mathcal{C}_\Gamma = 3$, and the minimum is attained exactly where the multi-term $a_{21}a_{52} + a_{31}a_{53}$ vanishes. On that locus the third column collapses onto e_4 and the fourth vanishes, so the realizations attaining $\dim \mathcal{C}_\Gamma$ intersect in $\operatorname{span}\{e_1, e_4\}$, while $\mathcal{C}_\Lambda^F = \operatorname{span}\{e_1, e_6\}$: neither of those two contains the other. Intersecting over the whole family takes both into account and leaves $\mathcal{C}_\Gamma^F = \operatorname{span}\{e_1\}$, inside $\mathcal{C}_\Lambda^F$ as (i) requires. The pattern is an input-connected HDAG whose single driver is its only source, so no hypothesis of this appendix recovers the restricted reading.

The same pattern shows why the restricted reading is tempting. It is the setting in which [12] computes the worst-case *dimension*, through a subgraph $\bar{\mathcal{G}}$ that keeps every node of $\mathcal{G}$ and drops every edge meeting a node at input-distance d or more (Definition 4.8), d being the first distance at which the nodes of the layer ℓ_d are all integrators or intermediators (Definition 4.30) whose multi-terms can be set to zero together. Here $d = 4$, so $\bar{\mathcal{G}}$ drops the edge $5 \to 6$, and its maximum disjoint stem-and-cycle covers are the single stems $u \to 1 \to 2 \to 4$, $u \to 1 \to 2 \to 5$ and $u \to 1 \to 3 \to 5$, three state nodes each: the reduction delivers $\dim \mathcal{C}_\Gamma = 3$ correctly (Theorem 6.4). It does not deliver a subspace. Node 4 lies on only one of those three stems, yet e_4 lies in the controllable subspace of every realization attaining $\dim \mathcal{C}_\Gamma$. That intersection is therefore not read off the reduction, and it is not the object the fixed subspaces compare.

Source and notation. The four-subspace framework, the containment of Fig. 6.2, and the subspace relation $\mathcal{C}_\Gamma^{\mathcal{F}} \subseteq \mathcal{C}_\Lambda^{\mathcal{F}}$ (the book's $\mathcal{C}_\Gamma^{F} \subseteq \mathcal{C}_\Lambda^{F}$, clause (i)) are [12]'s. Its node-level form, "every FSSC node is an FSC node", is a theorem of the author's [1]. The book takes $\mathcal{C}_\Gamma^{F}$ in the sense the source states for it (controllability of a direction at every realization), and under that reading the relation is the one-line argument of (i) above. A wording difference is recorded rather than resolved: the source's Definition 1 words $\mathcal{C}_\Gamma^{\mathcal{F}}$ as an intersection taken over the realizations attaining $\dim \mathcal{C}_\Gamma$, which names the other subspace of the paragraphs above, and for that one the relation does not hold on Remark 8.5's pattern. Separately, the source's Theorem 1 tests FSSC membership by comparing $\dim \mathcal{C}_\Gamma$ before and after the input addition. The book keeps that comparison as a necessary condition only (Theorem 8.7 and Remark 8.8), the exact criterion being Lemma B.3 and clause (ii). Notation translated as: the source's subgraph $\bar{\mathcal{G}}(T_c)$, written $\bar{\mathcal{G}}(\mathcal{V}, \bar{\mathcal{E}})$ in the published paper, is $\bar{\mathcal{G}}$, and "leader" is driver node. The disjoint stem-and-cycle characterization of the SC-side test is Hosoe's theorem, the book's Theorem 6.4.

Bibliography

[1] N.-J. Park, "Controllability and controllable subspaces of structured networks: A graph-theoretical approach," Ph.D. dissertation, Gwangju Institute of Science and Technology, 2024.

[2] C.-T. Chen, *Linear System Theory and Design*, 3rd ed. New York: Oxford University Press, 1999.

[3] Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási, "Controllability of complex networks," *Nature*, vol. 473, no. 7346, pp. 167–173, 2011.

[4] N.-J. Park and H.-S. Ahn, "Network controllability of structured networks: A survey on graph-theoretical approaches," 2026, manuscript under review, SICE JCMSI.

[5] J. P. Hespanha, *Linear Systems Theory*, 2nd ed. Princeton University Press, 2018.

[6] H. K. Khalil, *Nonlinear systems*. Prentice Hall, 2002.

[7] C. Godsil and G. F. Royle, *Algebraic graph theory*, ser. Graduate Texts in Mathematics. New York: Springer, 2001, vol. 207.

[8] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1520–1533, 2004.

[9] W. Ren and R. W. Beard, "Consensus seeking in multiagent systems under dynamically changing interaction topologies," *IEEE Transactions on Automatic Control*, vol. 50, no. 5, pp. 655–661, 2005.

[10] J.-M. Dion, C. Commault, and J. V. der Woude, "Generic properties and control of linear structured systems: a survey," *Automatica*, vol. 39, no. 7, pp. 1125–1144, 2003.

[11] N.-J. Park, Y.-B. Bae, K. L. Moore, and H.-S. Ahn, "Controllability matrix analysis of structured networks: A tight lower bound on the dimension of controllable subspace," in *2024 European Control Conference (ECC)*, 2024, pp. 1165–1170.

[12] N.-J. Park, Y.-U. Kim, K.-H. Oh, and H.-S. Ahn, "Controllable subspaces in structured networks of hierarchical directed acyclic graphs: Controllability of individual nodes," *IEEE Transactions on Automatic Control*, vol. 71, no. 4, pp. 2630–2636, 2026.

[13] J. Gao, Y.-Y. Liu, R. M. D'souza, and A.-L. Barabási, "Target control of complex networks," *Nature Communications*, vol. 5, no. 1, p. 5415, 2014.

[14] C. Commault, J. van der Woude, and P. Frasca, "Functional target controllability of networks: Structural properties and efficient algorithms," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 3, pp. 1521–1530, 2019.

[15] J. Li, X. Chen, S. Pequito, G. J. Pappas, and V. M. Preciado, "On the structural target controllability of undirected networks," *IEEE Transactions on Automatic Control*, vol. 66, no. 10, pp. 4836–4843, 2021.

[16] C.-T. Lin, "Structural controllability," *IEEE Transactions on Automatic Control*, vol. 19, no. 3, pp. 201–208, 1974.

[17] H. Mayeda and T. Yamada, "Strong structural controllability," *SIAM Journal on Control and Optimization*, vol. 17, no. 1, pp. 123–138, 1979.

[18] S. Poljak, "On the generic dimension of controllable subspaces," *IEEE Transactions on Automatic Control*, vol. 35, no. 3, pp. 367–369, 1990.

[19] S. Hosoe, "Determination of generic dimensions of controllable subspaces and its application," *IEEE Transactions on Automatic Control*, vol. 25, no. 6, pp. 1192–1196, 1980.

[20] A. Y. Yazicioglu, M. Shabbir, W. Abbas, and X. Koutsoukos, "Strong structural controllability of diffusively coupled networks: Comparison of bounds based on distances and zero forcing," in *Proceedings of the IEEE Conference on Decision and Control (CDC)*, 2020, pp. 566–571.

[21] A. Y. Yazicioglu, W. Abbas, and M. Egerstedt, "Graph distances and controllability of networks," *IEEE Transactions on Automatic Control*, vol. 61, no. 12, pp. 4125–4130, 2016.

[22] R. W. Shields and J. B. Pearson, "Structural controllability of multiinput linear systems," *IEEE Transactions on Automatic Control*, vol. 21, no. 2, pp. 203–212, 1976.

[23] N.-J. Park, S.-H. Kwon, Y.-B. Bae, B.-Y. Kim, K. L. Moore, and H.-S. Ahn, "Merging rules for strong structural controllability and minimum input problem in undirected networks," *Journal of the Franklin Institute*, vol. 362, p. 107589, 2025.

[24] G. Reissig, C. Hartung, and F. Svaricek, "Strong structural controllability and observability of linear time-varying systems," *IEEE Transactions on Automatic Control*, vol. 59, no. 11, pp. 3087–3092, 2014.

[25] N. Monshizadeh, S. Zhang, and M. K. Camlibel, "Zero forcing sets and controllability of dynamical systems defined on graphs," *IEEE Transactions on Automatic Control*, vol. 59, no. 9, pp. 2562–2567, 2014.

[26] M. Trefois and J.-C. Delvenne, "Zero forcing number, constrained matchings and strong structural controllability," *Linear Algebra and its Applications*, vol. 484, pp. 199–218, 2015.

[27] S. S. Mousavi, M. Haeri, and M. Mesbahi, "On the structural and strong structural controllability of undirected networks," *IEEE Transactions on Automatic Control*, vol. 63, no. 7, pp. 2234–2241, 2018.

[28] N.-J. Park, Y.-U. Kim, and H.-S. Ahn, "Strong structural controllability of directed graphs via zero forcing sets," *IFAC-PapersOnLine*, vol. 56, no. 2, pp. 3441–3446, 2023.

[29] C. Commault, J. V. der Woude, and T. Boukhobza, "On the fixed controllable subspace in linear structured systems," *Systems & Control Letters*, vol. 102, pp. 42–47, 2017.

[30] N.-J. Park, Y.-U. Kim, and H.-S. Ahn, "Fixed node determination and analysis in directed acyclic graphs of structured networks," *Journal of the Franklin Institute*, vol. 361, p. 106995, 2024.

[31] N.-J. Park, Y.-U. Kim, H.-S. Ahn, and K. L. Moore, "Strong structural controllability and observability of individual nodes: A graph-theoretical approach," *IEEE Transactions on Automatic Control*, vol. 69, no. 4, pp. 2598–2604, 2024.

[32] C. Commault, J. V. der Woude, and T. Boukhobza, "A dynamic graph characterisation of the fixed part of the controllable subspace of a linear structured system," *Systems & Control Letters*, vol. 129, pp. 17–25, 2019.

[33] A. Olshevsky, "Minimal controllability problems," *IEEE Transactions on Control of Network Systems*, vol. 1, no. 3, pp. 249–258, 2014.

[34] F. Barioli, W. Barrett, S. M. Fallat, H. T. Hall, L. Hogben, B. Shader, P. V. den Driessche, and H. V. der Holst, "Zero forcing parameters and minimum rank problems," *Linear Algebra and its Applications*, vol. 433, no. 2, pp. 401–411, 2010.

[35] S. Gu, F. Pasqualetti, M. Cieslak, Q. K. Telesford, A. B. Yu, A. E. Kahn, J. D. Medaglia, J. M. Vettel, M. B. Miller, and S. T. Grafton, "Controllability of structural brain networks," *Nature Communications*, vol. 6, no. 1, p. 8414, 2015.

[36] F. Pasqualetti, S. Zampieri, and F. Bullo, "Controllability metrics, limitations and algorithms for complex networks," *IEEE Transactions on Control of Network Systems*, vol. 1, no. 1, pp. 40–52, 2014.

[37] I. Klickstein, A. Shirin, and F. Sorrentino, "Energy scaling of targeted optimal control of complex networks," *Nature Communications*, vol. 8, no. 1, p. 15145, 2017.

[38] S. Pequito, S. Kar, and A. P. Aguiar, "A framework for structural input/output and control configuration selection in large-scale systems," *IEEE Transactions on Automatic Control*, vol. 61, no. 2, pp. 303–318, 2016.

[39] T. H. Summers, F. L. Cortesi, and J. Lygeros, "On submodularity and controllability in complex dynamical networks," *IEEE Transactions on Control of Network Systems*, vol. 3, no. 1, pp. 91–101, 2016.

[40] K. Fitch and N. E. Leonard, "Optimal leader selection for controllability and robustness in multi-agent networks," in *Proceedings of the European Control Conference (ECC)*, 2016, pp. 1550–1555.

Index