Chapter 2
Algebraic Graph Theory and Cooperative Control Consensus

Cooperative control studies the dynamics of multi-agent dynamical systems linked to each other by a communication graph. The graph represents the allowed information flow between the agents. The objective of cooperative control is to devise control protocols for the individual agents that guarantee synchronized behavior of the states of all the agents in some prescribed sense. In cooperative systems, any control protocol must be distributed in the sense that it respects the prescribed graph topology. That is, the control protocol for each agent is allowed to depend only on information about that agent and its neighbors in the graph. The communication restrictions imposed by graph topologies can severely limit what can be accomplished by local distributed control protocols at each agent. In fact, the graph topological properties complicate the design of synchronization controllers and result in intriguing behaviors of multi-agent systems on graphs that do not occur in single-agent, centralized, or decentralized feedback control systems.

Of fundamental concern for cooperative systems on graphs is the study of their collective behaviors under the influence of the information flow allowed in the graph. This chapter introduces cooperative synchronization control of multi-agent dynamical systems interconnected by a fixed communication graph topology. Each agent or node is mathematically modeled by a dynamical linear time-invariant (LTI) system. First, in Sect. 2.1 we give a review of graph basics and algebraic graph theory, which studies certain matrices associated with the graph. The graph Laplacian matrix is introduced and its eigenstructure is studied, including the first eigenvalue, the first left eigenvector, and the second eigenvalue or Fiedler eigenvalue.

In Sect. 2.2, dynamical systems on graphs are introduced. The idea of distributed control and the consensus problem are introduced. Section 2.3 studies consensus for first-order integrator dynamics for continuous-time systems. The basic results and key ideas of cooperative control emerge from this study. The importance of the graph eigenstructure in interconnected agent dynamical behavior is shown. Section 2.4 reveals the existence of certain motion invariants for consensus on graphs. Section 2.5 studies consensus for first-order discrete-time systems.

Section 2.6 introduces consensus for general linear dynamical systems on graphs. A fundamental result is given that relates the synchronization of multi-agent systems to the stability of a set of systems that depend on the graph topology prop-

F. L. Lewis et al., *Cooperative Control of Multi-Agent Systems,*
Communications and Control Engineering, DOI 10.1007/978-1-4471-5574-4_2,

erties. This reveals the relationships between local agent feedback design and the global synchronization properties based on the graph communication restrictions. Section 2.7 studies consensus for second-order position–velocity systems which include motion control in formations.

In Sect. 2.8, we present some key concepts needed in this book for the design of optimal and adaptive distributed controllers on graphs. The section introduces several important matrix analysis methods for systems in graphs, including irreducible matrices, Frobenius form, stochastic matrices, and M-matrices. Relationships with the graph topology and eigenstructure are given.

In Sect. 2.9, we introduce Lyapunov functions for the analysis of the stability properties of cooperative multi-agent control systems. It is seen that Lyapunov functions for studying the stability properties of cooperative control protocols depend on the graph topology. Therefore, a given cooperative control protocol may be stable on one graph topology but not on another. This reveals the close interactions between the performance of locally designed control protocols and the manner in which the agents are allowed to communicate.

This chapter follows the development of cooperative control results in the early literature since the first papers [7, 9, 10, 15, 22]. To understand the relationships between the communication graph topology and the local design of distributed feedback control protocols, it was natural to study first-order systems and then second-order systems. Early applications were made to formation control [13, 14, 16, 18]. These studies brought an understanding of the limitations and caveats imposed by the graph communication restrictions and opened the way for many well-known results from systems theory to be extended to the case of multi-agent systems on graphs.

The relations discovered between the communication graph topology and the design of distributed control protocols have resulted in new design techniques for cooperative feedback control. New intriguing interactions have been discovered between graph topology and local control protocol design that reveal the richness of the study of cooperative multi-agent control on graphs, where new phenomena are seen that do not occur in control of single-agent systems. In the remainder of the book, we explore these relationships for optimal design and adaptive control on graphs.

2.1 Algebraic Graph Theory

In this book, we are concerned with the behaviors and interactions of dynamical systems that are interconnected by the links of a communication network. This communication network is modeled as a graph with directed edges corresponding to the allowed flow of information between the systems. The systems are modeled as the nodes in the graph and are sometimes called agents. We call this the study of multi-agent dynamical systems on graphs. The fundamental control issues concern how the graph topology interacts with the local feedback control protocols of the agents to produce overall behaviors of the interconnected nodes.

2.1.1 Graph Theory Basics

Here we present some basic graph theory concepts that are essential in the study of multi-agent dynamical systems. Good references are [3, 5].

Basic Definitions and Connectivity

A *graph* is a pair $G=(V,E)$ with $V=\{v_1,\cdots,v_N\}$ being a set of N nodes or vertices and E a set of edges or arcs. Elements of E are denoted as (v_i,v_j) which is termed an edge or arc from v_i to v_j, and represented as an arrow with tail at v_i and head at v_j. We assume the graph is simple, i.e., $(v_i,v_i)\notin E,\forall i$ no self-loops, and no multiple edges between the same pairs of nodes. Edge (v_i,v_j) is said to be outgoing with respect to node v_i and incoming with respect to v_j; and node v_i is termed the parent and v_j the child. The in-degree of v_i is the number of edges having v_i as a head. The out-degree of a node v_i is the number of edges having v_i as a tail. The set of (in-) neighbors of a node v_i is $N_i=\{v_j:(v_j,v_i)\in E\}$, i.e., the set of nodes with edges incoming to v_i. The number of neighbors $|N_i|$ of node v_i is equal to its in-degree.

If the in-degree equals the out-degree for all nodes $v_i\in V$ the graph is said to be *balanced*. If $(v_i,v_j)\in E\Rightarrow(v_j,v_i)\in E,\forall i,j$ the graph is said to be *bidirectional*, otherwise it is termed a directed graph or digraph. Associate with each edge $(v_j,v_i)\in E$ a weight a_{ij}. Note the order of the indices in this definition. We assume in this chapter that the nonzero weights are strictly positive. A graph is said to be *undirected* if $a_{ij}=a_{ji},\forall i,j$, that is, if it is bidirectional and the weights of edges (v_i,v_j) and (v_j,v_i) are the same.

A directed path is a sequence of nodes $v_0,v_1,\cdots,v_r$ such that $(v_i,v_{i+1})\in E$, $i\in\{0,1,\cdots,r-1\}$. Node v_i is said to be connected to node v_j if there is a directed path from v_i to v_j. The distance from v_i to v_j is the length of the shortest path from v_i to v_j. Graph G is said to be strongly connected if v_i,v_j are connected for all distinct nodes $v_i,v_j\in V$. For bidirectional and undirected graphs, if there is a directed path from v_i to v_j, then there is a directed path from v_j to v_i, and the qualifier 'strongly' is omitted.

A (directed) tree is a connected digraph where every node except one, called the root, has in-degree equal to one. A spanning tree of a digraph is a directed tree formed by graph edges that connects all the nodes of the graph. A graph is said to have a spanning tree if a subset of the edges forms a directed tree. This is equivalent to saying that all nodes in the graph are reachable from a single (root) node by following the edge arrows. A graph may have multiple spanning trees. Define the root set or leader set of a graph as the set of nodes that are the roots of all spanning trees.

If a graph is strongly connected, it contains at least one spanning tree. In fact, if a graph is strongly connected, then all nodes are root nodes.

2.1.2 Graph Matrices

Graph structure and properties can be studied by examining the properties of certain matrices associated with the graph. This is known as algebraic graph theory [3, 5].

Given the edge weights a_{ij}, a graph can be represented by an adjacency or connectivity matrix $A = [a_{ij}]$ with weights $a_{ij} > 0$ *if* $(v_j, v_i) \in E$ and $a_{ij} = 0$ otherwise. Note that $a_{ii} = 0$. Define the weighted in-degree of node v_i as the i-th row sum of A

$$d_i = \sum_{j=1}^{N} a_{ij} \tag{2.1}$$

and the weighted out-degree of node v_i as the i-th column sum of A

$$d_i^o = \sum_{j=1}^{N} a_{ji} \tag{2.2}$$

The in-degree and out-degree are local properties of the graph. Two important global graph properties are the diameter *Diam G*, which is the greatest distance between two nodes in a graph, and the (in-)volume, which is the sum of the in-degrees

$$Vol\, G = \sum_i d_i \tag{2.3}$$

The adjacency matrix A of an undirected graph is symmetric, $A = A^T$. A graph is said to be *weight balanced* if the weighted in-degree equals the weighted out-degree for all i. If all the nonzero edge weights are equal to 1, this is the same as the definition of balanced graph. An undirected graph is weight balanced, since if $A = A^T$ then the ith row sum equals the ith column sum. We may be loose at times and refer to node v_i simply as node i, and refer simply to in-degree, out-degree, and the balanced property, without the qualifier 'weight', even for graphs having non-unity weights on the edges.

Graph Laplacian Matrix Define the diagonal in-degree matrix $D = diag\{d_i\}$ and the (weighted) graph Laplacian matrix $L = D - A$. Note that L has all row sums equal to zero. Many properties of a graph may be studied in terms of its graph Laplacian. In fact, we shall see that the Laplacian matrix is of extreme importance in the study of dynamical multi-agent systems on graphs.

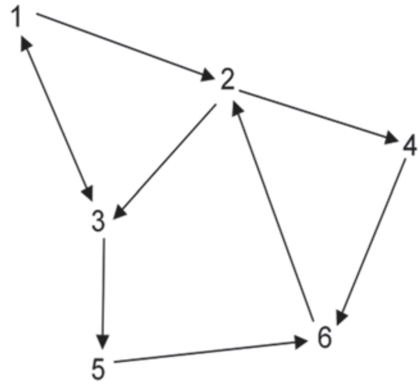

Fig. 2.1 A directed graph

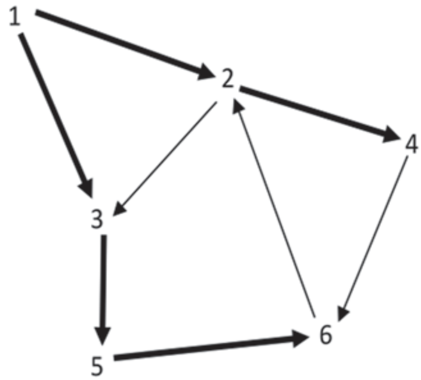

Fig. 2.2 A spanning tree for the graph in Fig. 2.1 with root node 1

Example 2.1 Graph Matrices

Consider the digraph shown in Fig. 2.1 with all edge weights equal to 1. The graph is strongly connected since there is a path between any two pairs of nodes. A spanning tree with root node 1 is shown in bold in Fig. 2.2. There are other spanning trees in this graph. In fact, every node is a root node since the graph is strongly connected.

The adjacency matrix is given by

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

and the diagonal in-degree matrix and Laplacian are

$$D=\begin{bmatrix}1&0&0&0&0&0\\0&2&0&0&0&0\\0&0&2&0&0&0\\0&0&0&1&0&0\\0&0&0&0&1&0\\0&0&0&0&0&2\end{bmatrix},\quad L=\begin{bmatrix}1&0&-1&0&0&0\\-1&2&0&0&0&-1\\-1&-1&2&0&0&0\\0&-1&0&1&0&0\\0&0&-1&0&1&0\\0&0&0&-1&-1&2\end{bmatrix}$$

Note that the row sums of L are all zero.

2.1.3 Eigenstructure of Graph Laplacian Matrix

We shall see that the eigenstructure of the graph Laplacian matrix L plays a key role in the analysis of dynamical systems on graphs. Define the Jordan normal form [2] of the graph Laplacian matrix by

$$L = MJM^{-1} \tag{2.4}$$

with the Jordan form matrix and transformation matrix given as

$$J=\begin{bmatrix}\lambda_1&&&\\&\lambda_2&&\\&&\ddots&\\&&&\lambda_N\end{bmatrix},\quad M=\begin{bmatrix}v_1&v_2&\cdots&v_N\end{bmatrix} \tag{2.5}$$

where the eigenvalues λ_i and right eigenvectors v_i satisfy

$$(\lambda_i I - L)v_i = 0 \tag{2.6}$$

with I being the identity matrix.

In general, the λ_i in (2.5) are not scalars but are Jordan blocks of the form

$$\begin{bmatrix}\lambda_i&1&&\\&\lambda_i&\ddots&\\&&\ddots&1\\&&&\lambda_i\end{bmatrix}$$

The number of such Jordan blocks associated with the same eigenvalue λ_i is known as the geometric multiplicity of eigenvalue λ_i. The sum of the sizes of all Jordan blocks associated with λ_i is called its algebraic multiplicity. For ease of notation and discussion, we assume here that the Jordan form is simple, that is, it is diagonal with all Jordan blocks of size 1. This is guaranteed if all eigenvalues of L are distinct. A symmetric matrix may not have distinct eigenvalues but has a simple Jordan form. All of our discussions generalize without difficulty to the case of nontrivial Jordan blocks.

The inverse of the transformation matrix M is given as

$$M^{-1} = \begin{bmatrix} w_1^T \\ w_2^T \\ \vdots \\ w_N^T \end{bmatrix} \tag{2.7}$$

where the left eigenvectors w_i satisfy

$$w_i^T(\lambda_i I - L) = 0 \tag{2.8}$$

and are normalized so that $w_i^T v_i = 1$.

We assume the eigenvalues are ordered so that $|\lambda_1| \le |\lambda_2| \le \cdots \le |\lambda_N|$. Any undirected graph has $L = L^T$ so all its eigenvalues are real and one can order them as $\lambda_1 \le \lambda_2 \le \cdots \le \lambda_N$.

Since L has all row sums zero, one has

$$L\underline{1}c = 0 \tag{2.9}$$

with $\underline{1} = [1 \cdots 1]^T \in R^N$ being the vector of ones and c any constant. Therefore, $\lambda_1 = 0$ is an eigenvalue with a right eigenvector of $\underline{1}c$. That is, $\underline{1}c \in N(L)$ the null-space of L. If the dimension of the null-space of L is equal to one, i.e., the rank of L is N-1, then $\lambda_1 = 0$ is nonrepeated and $\underline{1}c$ is the only vector in $N(L)$. The next standard result states when this occurs.

Theorem 2.1 *L has rank N-1, i.e.,* $\lambda_1 = 0$ *is nonrepeated, if and only if graph G has a spanning tree* [13, 16].

If the graph has a spanning tree, then $|\lambda_2| > 0$. If the graph is strongly connected, then it has a spanning tree and L has rank N-1.

The Laplacian has at least one eigenvalue at $\lambda_1 = 0$. The remaining eigenvalues can be localized using the following result.

Geršgorin Circle Criterion All eigenvalues of a matrix $E = [e_{ij}] \in R^{N \times N}$ are located within the union of N disks [6].

$$\bigcup_{i=1}^{N} \left\{ z \in C : |z - e_{ii}| \le \sum_{j \ne i} |e_{ij}| \right\}$$

The i-th disk in the Geršgorin circle criterion is drawn with a center at the diagonal element e_{ii} and with a radius equal to the i-th absolute row sum with the diagonal element deleted, $\sum_{j \ne i} |e_{ij}|$. Therefore, the Geršgorin disks for the graph Laplacian matrix $L = D - A$ are centered at the in-degrees d_i and have radius equal to d_i. Let d_{Max} be the maximum in-degree of G. Then, the largest Geršgorin disk of the

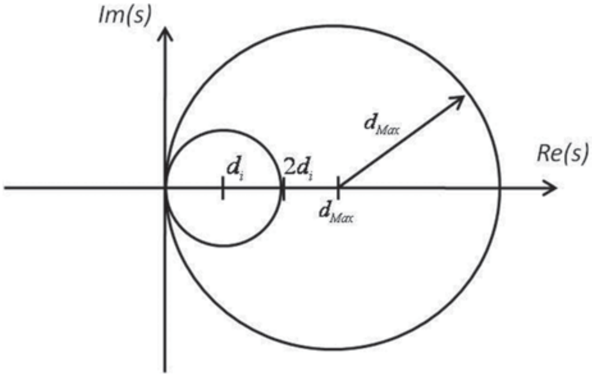

Fig. 2.3 Geršgorin disks of L in the complex plane

Laplacian matrix L is given by a circle centered at d_{Max} and having radius of d_{Max}. This circle contains all the eigenvalues of L. See Fig. 2.3. The Geršgorin circle criterion ties the eigenvalues of L rather closely to the graph structural properties in terms of the in-degrees.

We have thus discovered that if the graph has a spanning tree, there is a nonrepeated eigenvalue at $\lambda_1 = 0$ and all other eigenvalues have positive real parts, i.e., in the open right-half plane, and are within a circle centered at d_{Max} and having radius of d_{Max}.

When comparing eigenvalues between two graphs, it is often more useful to use the normalized Laplacian matrix

$$\bar{L} = D^{-1}L = D^{-1}(D - A) = I - D^{-1}A \tag{2.10}$$

Since the normalized adjacency matrix $\bar{A} = D^{-1}A$ has row sums equal to one, $d_i = 1, \forall i$ and $\bar{L}$ has all Geršgorin disks centered at $s = 1$ with radius of 1.

Example 2.2 Laplacian Matrix Eigenvalues for Various Graph Types

In this example, we will compute the eigenvalues of the graph Laplacian matrix $L = D - A$ for various types of graphs. The intent is to give a feeling for the dependence of the Laplacian eigenvalues on the graph topology. This example is the work of David Maxwell in the class EE 5329 Distributed Decision & Control, Spring 2011, Department of Electrical Engineering, The University of Texas at Arlington, TX, USA. The class link is http://arri.uta.edu/acs

The graphs studied in this example include several commonly occurring topologies and are depicted in Fig. 2.4. The graph notation is standard and is explained in [3, 5]. We include it for interest only and do not use this notation further in the book. All edge weights are taken equal to 1. Note that a complete (or fully connected) graph is one that has all possible edges between the nodes.

The eigenvalues of these graphs are listed in Table 2.1 and plotted in the complex s-plane in Fig. 2.5. Several things are worthy of note:

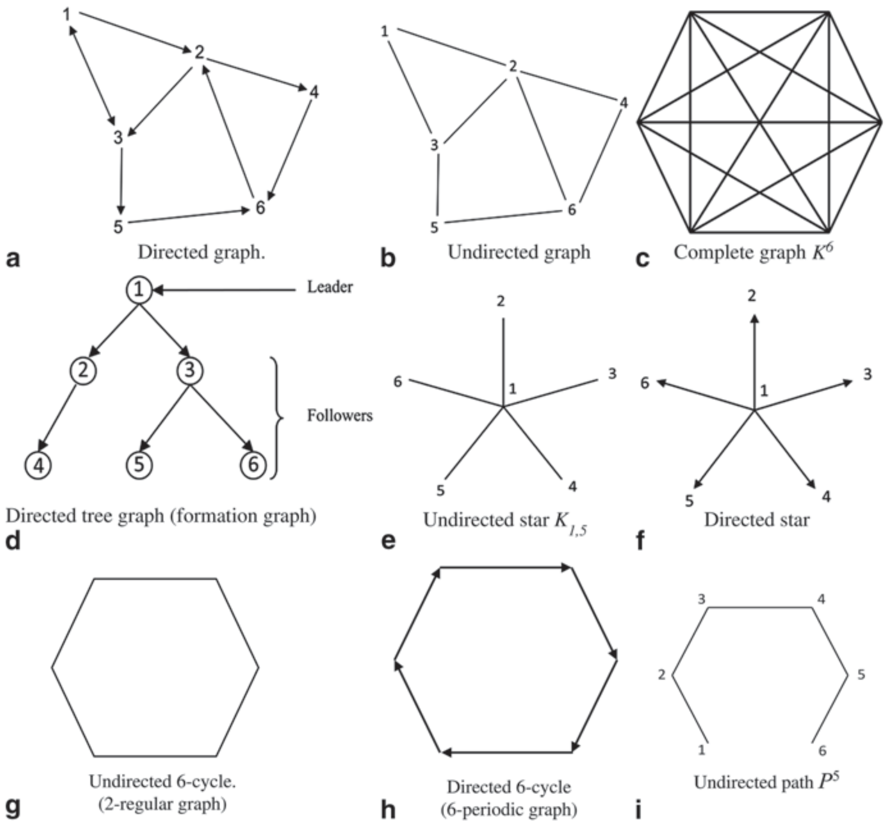

Fig. 2.4 Different graph topologies for Example 2.2. **a** Directed graph. **b** Undirected graph. **c** Complete graph K^6. **d** Directed tree graph (formation graph). **e** Undirected star $K_{1,5}$. **f** Directed star. **g** Undirected 6-cycle (2-regular graph). **h** Directed 6-cycle (6-periodic graph). **i** Undirected path P^5

Table 2.1 Eigenvalues for different graph topologies in Example 2.2

a. Directed graph	b. Undirected graph	c. Complete graph	d. Directed tree	e. Undirected star	f. Directed star	g. Undirected 6-cycle	h. Directed 6-cycle	i. Undirected path
0	0	0	0	0	0	0	0.0000	0
0.7793	1.3820	6	1	1	1	1	0.5 + 0.866i	0.2679
1.0000	1.6972	6	1	1	1	1	0.5 − 0.866i	1
2.2481 + 1.0340i	3.6180	6	1	1	1	3	1.5 + 0.866i	2
2.2481 − 1.0340i	4.0000	6	1	1	1	3	1.5 − 0.866i	3
2.7245	5.3028	6	1	6	1	4	2.	3.7321

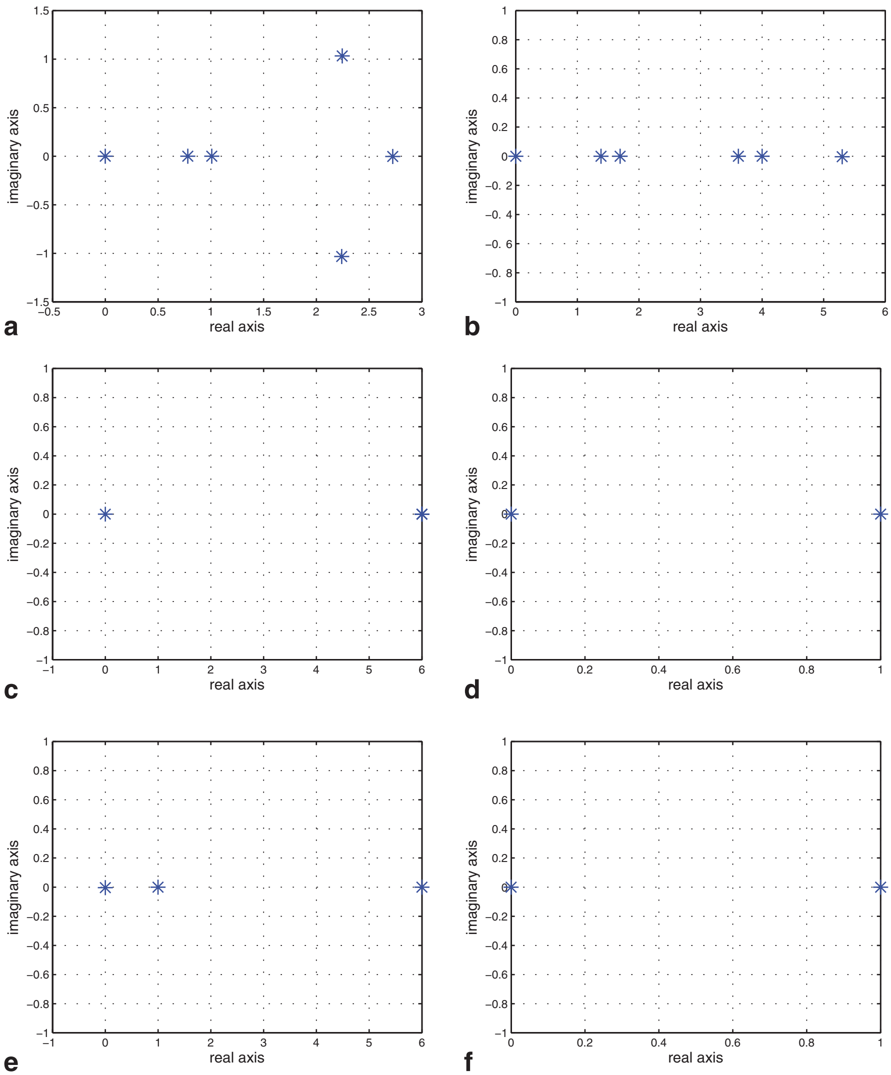

Fig. 2.5 Complex plane plots of graph eigenvalues for Example 2.2. **a** Directed graph. **b** Undirected graph. **c** Complete graph K^6. **d** Directed tree (formation graph). **e** Undirected star $K_{1,5}$. **f** Directed star. **g** Undirected 6-cycle (2-regular graph). **h** Directed 6-cycle (6-periodic graph). **i** Undirected path P^5

1. The Laplacian matrix has row sums equal to zero so that all graphs have the first eigenvalue at $\lambda_1 = 0$.
2. All undirected graphs have a symmetric Laplacian matrix L and so their graph eigenvalues are real.

http://www.springer.com/978-1-4471-5573-7

Cooperative Control of Multi-Agent Systems
Optimal and Adaptive Design Approaches
Lewis, F.L.; Zhang, H.; Hengster-Movric, K.; Das, A.
2014, XX, 307 p. 80 illus., 59 illus. in color., Hardcover
ISBN: 978-1-4471-5573-7