

21

Other Matrices Used in Graph

其他矩阵

关联矩阵、度矩阵、拉普拉斯矩阵等等



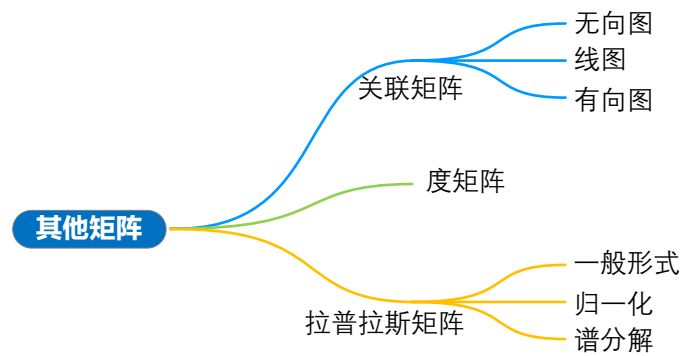
一切真理都经过三个阶段：首先，被讥讽嘲笑；然后，被强烈反对；最后，它被认为是不言而喻，不辩自明。

All truth passes through three stages: First, it is ridiculed; Second, it is violently opposed; Third, it is accepted as self-evident.

—— 阿图尔·叔本华 (Arthur Schopenhauer) | 德国哲学家 | 1788 ~ 1860



- ◀ `networkx.adjacency_matrix()` 计算图的邻接矩阵
- ◀ `networkx.incidence_matrix()` 计算图的关联矩阵
- ◀ `networkx.line_graph()` 将无向图转换为线图
- ◀ `networkx.laplacian_matrix()` 计算无向图的拉普拉斯矩阵
- ◀ `networkx.normalized_laplacian_matrix()` 计算无向图的归一化拉普拉斯矩阵
- ◀ `networkx.laplacian_spectrum()` 无向图的拉普拉斯矩阵谱分析
- ◀ `networkx.normalized_laplacian_spectrum()` 无向图的归一化拉普拉斯矩阵谱分析



21.1 图中常见矩阵

前文介绍了和图直接相关的**邻接矩阵** (adjacency matrix)，本章则介绍如下几个和图相关的矩阵。

- ▶ **关联矩阵** (incidence matrix)：关联矩阵是另一种表示图的矩阵方式，它描述了图中节点和边之间的关系。如果有 n 个节点和 m 条边，那么关联矩阵的大小为 $n \times m$ 。矩阵中的元素 a_{ij} 表示节点 i 和边 j 之间的关系。
- ▶ **度矩阵** (degree matrix)：度矩阵是一个对角矩阵，其对角元素表示每个节点的度数。
- ▶ **拉普拉斯矩阵** (Laplacian matrix)：一般值得的是度矩阵和邻接矩阵之差。

我们可以利用线性代数方法研究这些矩阵，进而解决图论中的问题，使得图的分析更加形式化和简化。

21.2 关联矩阵

在图论中，关联矩阵是一种表示图结构的矩阵。这个矩阵的行对应于图的节点集合，列对应于图的边集合。对于一个有 V 个节点和 E 条边的图，关联矩阵的大小为 $V \times E$ 。

对于无向图，矩阵中的元素表示节点和边之间的关系，通常使用 0 和 1 表示。具体而言，如果无向图中节点和边相关联，则对应的矩阵元素为 1；否则为 0。

对于有向图，那么关联矩阵的元素可能取值为 -1、0 和 1，正负号表示边的方向。

无向图

图 2 中这幅无自环无向图已经出现过很多次了，下面首先回顾它的邻接矩阵。

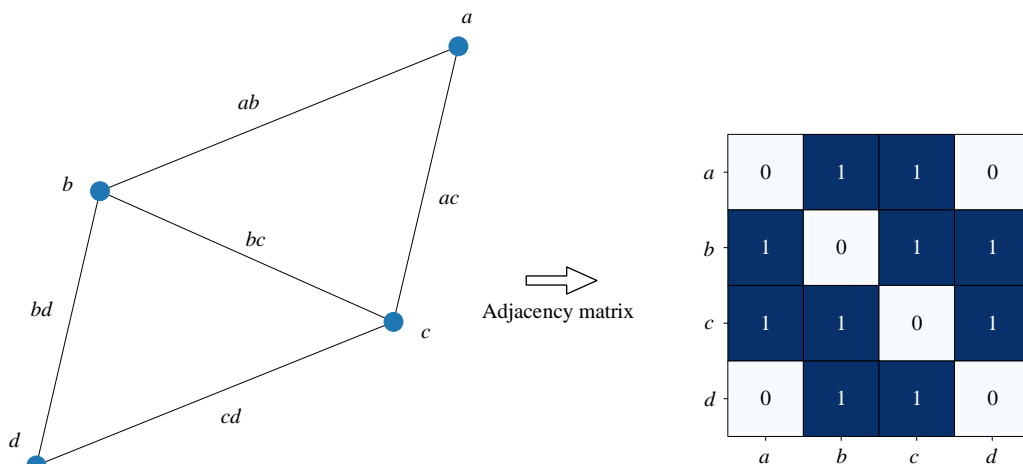


图 1. 无向图到邻接矩阵热图

这幅无向图的关联矩阵 C 为

$$C = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \quad (1)$$

如图2所示，这幅图有4个节点、5条边，因此其关联矩阵 C 的形状为 4×5 。关联矩阵 C 的4行从上到下分别代表4个节点—— a 、 b 、 c 、 d 。 C 的5列从左到右分别代表5条边—— ab 、 ac 、 bc 、 bd 、 cd 。

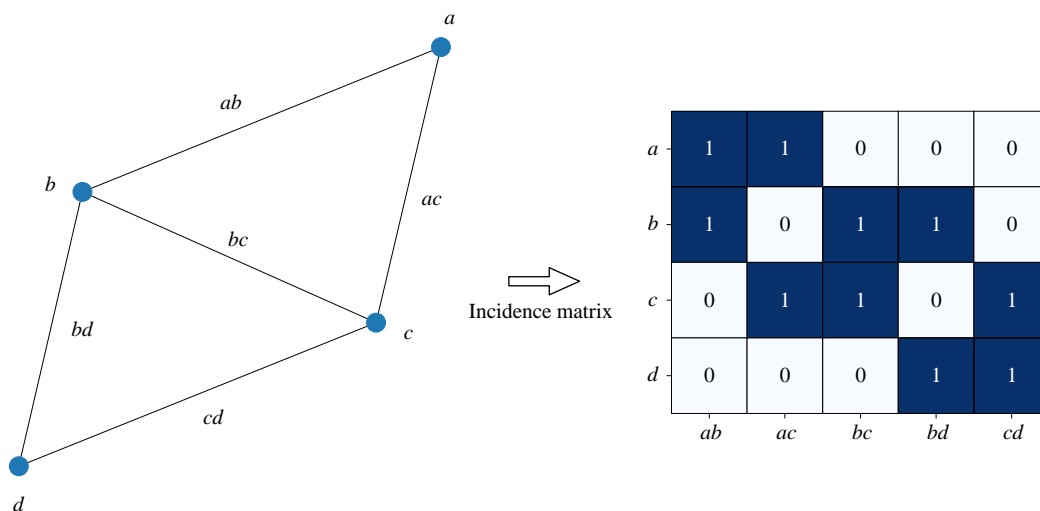


图2. 从无向图到关联矩阵热图

对于不含自环无向图，不考虑权重、不考虑多图的话，关联矩阵 C 每列元素之和为2，(1)中关联矩阵列元素之和为如下向量

$$\begin{bmatrix} 2 & 2 & 2 & 2 & 2 \end{bmatrix} \quad (2)$$

(1)中关联矩阵行元素之和则是每个节点的度

$$\begin{bmatrix} 2 \\ 3 \\ 3 \\ 2 \end{bmatrix} \quad (3)$$

图3逐个元素解释无自环无向图和关联矩阵之间的关系。比如，关联矩阵 C 的第1行代表和节点 a 有关的边，即 ab 、 ac ；因此， $c_{1,1}$ 和 $c_{1,2}$ 元素均为1。

请大家自己分析图3剩余子图。

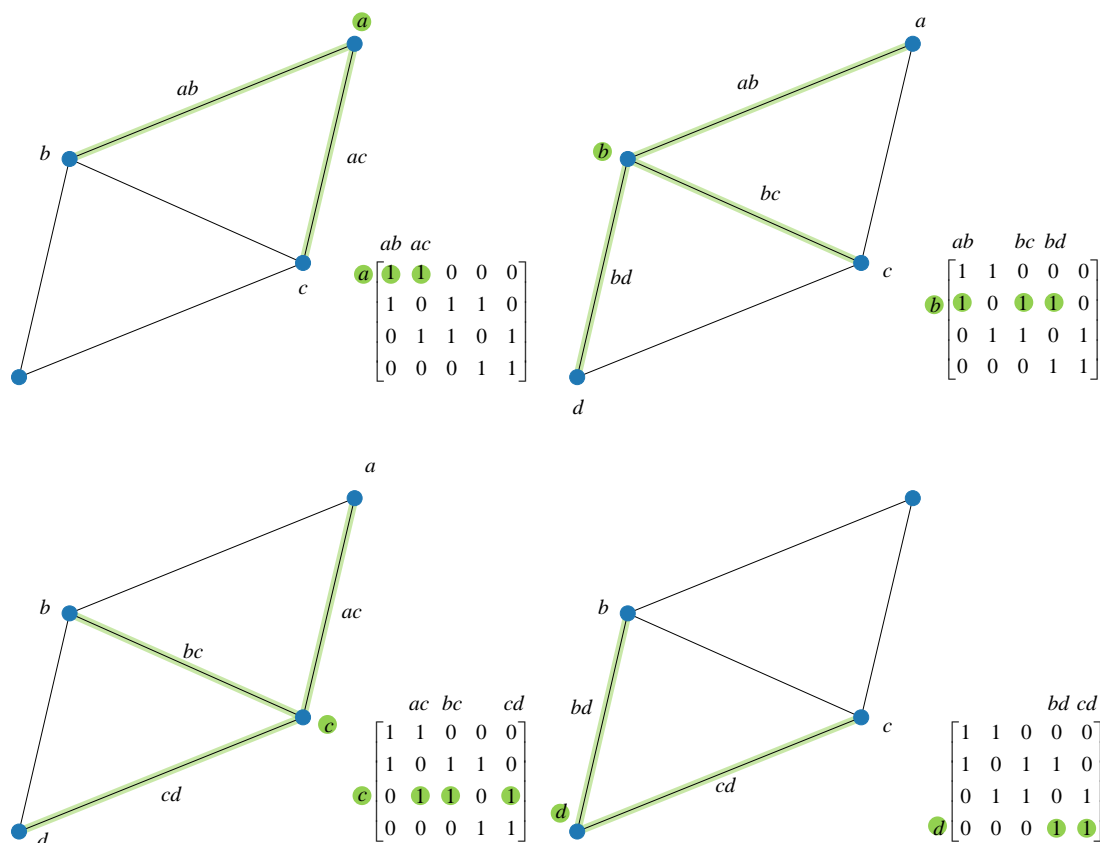


图 3. 逐个元素解释无自环无向图和关联矩阵之间的关系

代码 1 计算图的邻接矩阵、关联矩阵，并绘制图 1 和图 2 中热图。

- a** 用 `networkx.adjacency_matrix()` 计算图的邻接矩阵。
- b** 用 `seaborn.heatmap()` 可视化邻接矩阵。纵轴刻度标签 `yticklabels` 和横轴刻度标签 `xticklabels` 都是无向图的节点，即 `list(G.nodes)`。
- c** 用 `networkx.incidence_matrix()` 计算图的关联矩阵
- d** 用 `seaborn.heatmap()` 可视化关联矩阵。纵轴刻度标签 `yticklabels` 为无向图的节点，即 `list(G.nodes)`；横轴刻度标签 `xticklabels` 则是无向图的边，即 `list(G.edges)`。

表 1 给出几幅图和它们对应的关联矩阵供大家在 NetworkX 中练习。请大家注意表中不同关联矩阵列代表的边不同。

```

import matplotlib.pyplot as plt
import networkx as nx
import numpy as np
import seaborn as sns

G = nx.Graph()
# 创建无向图的实例

G.add_nodes_from(['a', 'b', 'c', 'd'])
# 添加多个顶点

G.add_edges_from([( 'a', 'b' ), ( 'b', 'c' ),
                  ( 'b', 'd' ), ( 'c', 'd' ),
                  ( 'c', 'a' )])

# 增加一组边


# 邻接矩阵
a A = nx.adjacency_matrix(G).todense()

# 可视化
b sns.heatmap(A, cmap = 'Blues',
              annot = True, fmt = '.0f',
              xticklabels = list(G.nodes),
              yticklabels = list(G.nodes),
              linecolor = 'k', square = True,
              linewidths = 0.2)
plt.savefig('邻接矩阵.svg')

c C = nx.incidence_matrix(G).todense()
# 关联矩阵

# 可视化
d sns.heatmap(C, cmap = 'Blues',
              annot = True, fmt = '.0f',
              yticklabels = list(G.nodes),
              xticklabels = list(G.edges),
              linecolor = 'k', square = True,
              linewidths = 0.2)
plt.savefig('关联矩阵.svg')

```

代码 1. 图的邻接矩阵和关联矩阵 |  Bk6_Ch21_01.ipynb

Bk6_Ch21_01.ipynb 还绘制空手道俱乐部人员关系图的关联矩阵热图，具体如图 4 所示。

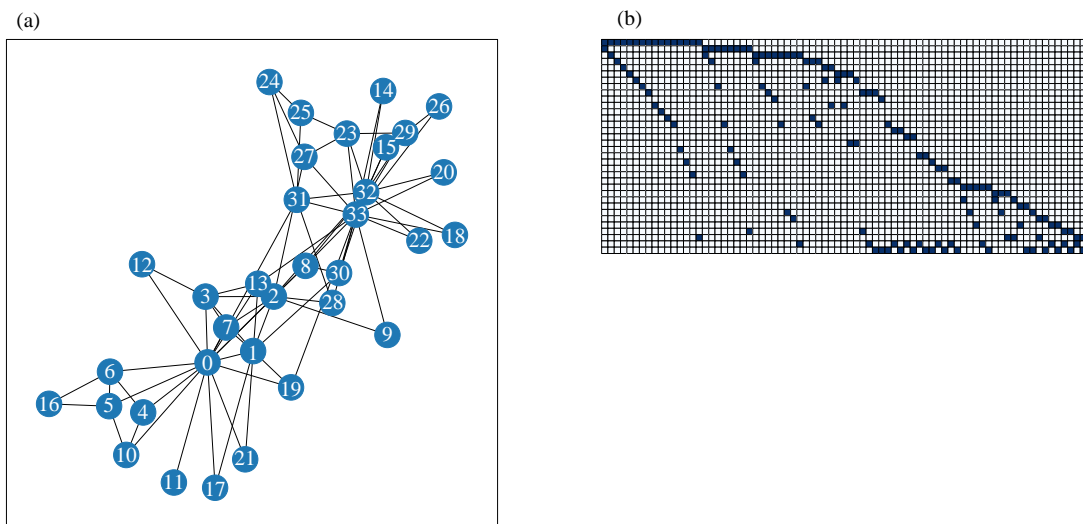


图 4. 空手道俱乐部人员关系图，以及对应关联矩阵热图

表 1.4 4 个节点构造的几种无向图及关联矩阵，不含自环，不加权

无向图	关联矩阵	无向图	关联矩阵
	$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \end{bmatrix}$		NA
	$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \end{bmatrix}$		$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}$
	$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{bmatrix}$		$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$
	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix}$		$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

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

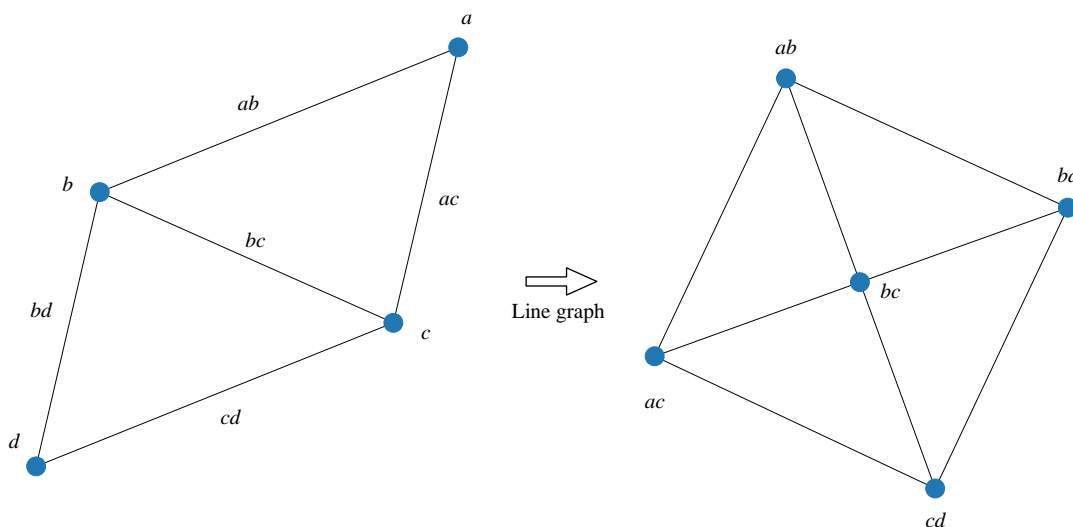
线图

下面，让我们聊聊一幅图的孪生兄弟——**线图** (line graph)。

图 G 自身的线图 $L(G)$ 是一张能够反映 G 各边邻接性的图， $L(G)$ 具体定义如下。

- ▶ $L(G)$ 的节点对应 G 的边。
- ▶ $L(G)$ 的节点相连，仅当它们在 G 中有公共节点。

白话来说，图 G 的边变成了线图 $L(G)$ 的节点；如果，两条边在图 G 通过公共节点相连，则它们在线图 $L(G)$ 中有一条边相连。



本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

图 5. 无向图与其线图

显然， $L(G)$ 也有自己的邻接矩阵 A_L ，如图 6 所示。

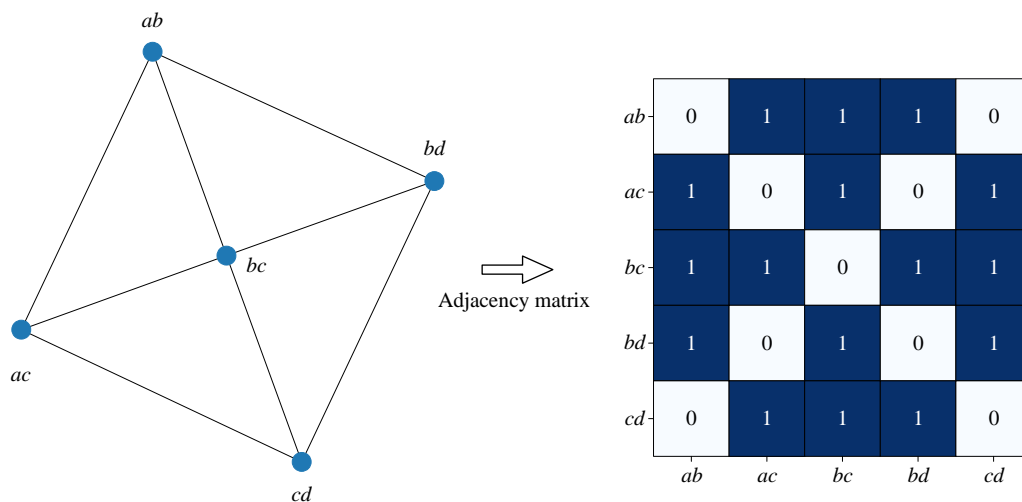


图 6. 线图的邻接矩阵

G 的线图邻接矩阵 A_L 和 G 的关联矩阵 C 存在如下关系

$$A_L = C^T C - 2I \quad (4)$$

其中， I 为单位矩阵，行、列数为图 G 的边数。具体计算过程如图 7 所示。

$$A_L = C^T @ C - 2I$$

0	1	1	1	0
1	0	1	0	1
1	1	0	1	1
1	0	1	0	1
0	1	1	1	0

1	1	0	0	0
1	0	1	0	0
0	1	1	0	0
0	1	0	1	0
0	0	1	1	1

1	1	0	0	0
1	0	1	1	0
0	1	1	0	1
0	1	0	1	1
0	0	0	1	1

2	0	0	0	0
0	2	0	0	1
0	0	2	0	0
0	0	0	2	0
0	0	0	0	2

图 7. 计算邻接矩阵 A_L

代码 2 将有向图转换为线图，并且验证 (4) 给出的关系。下面聊聊代码中关键语句。

```

import matplotlib.pyplot as plt
import networkx as nx
import numpy as np
import seaborn as sns

undirected_G = nx.Graph()
# 创建无向图的实例

undirected_G.add_nodes_from(['a', 'b', 'c', 'd'])
# 添加多个顶点

undirected_G.add_edges_from([('a', 'b'),
                              ('b', 'c'),
                              ('b', 'd'),
                              ('c', 'd'),
                              ('c', 'a')])

# 增加一组边

a sequence_edges_G = list(undirected_G.edges())
# 获取无向图边的序列，用于关联矩阵列排序

# 转换成线图
b L_G = nx.line_graph(undirected_G)

# 可视化线图
c plt.figure(figsize = (6,6))
nx.draw_networkx(L_G, pos = nx.spring_layout(L_G),
                 node_size = 180)
plt.savefig('线图.svg')

# 线图的链接矩阵
# 调整列顺序，对齐列，这样方便后续矩阵运算
d A_LG = nx.adjacency_matrix(L_G,
                             nodelist = sequence_edges_G).todense()

# 图的关联矩阵
e C = nx.incidence_matrix(undirected_G).todense()

# 验证矩阵关系
# A_LG = C.T @ C - 2*I
f C.T @ C - 2 * np.identity(5)

```

代码 2. 图线图 | Bk6_Ch21_02.ipynb

a 用 `edges()` 方法获取无向图边的排序，然后将其转换为列表；这个列表之后会用来重新排序关联矩阵的列。

b 用 `networkx.line_graph()` 将无向图转换为线图。

c 用 `networkx.draw_networkx()` 绘制线图。再次强调，线图也是一种图；只不过线图的节点是原始图的边。

d 用 `networkx.adjacency_matrix()` 获取线图的邻接矩阵。

参数 `nodelist = sequence_edges_G` 保证，线图邻接矩阵和关联矩阵的列对齐。

e 用 `networkx.incidence_matrix()` 获取原始图的关联矩阵。

f 给出的矩阵运算用来验证 (4)。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

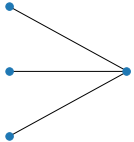
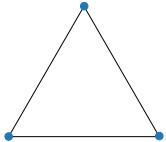
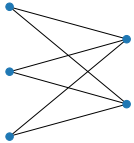
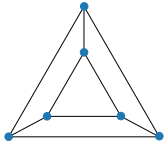
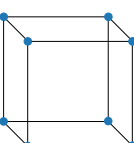
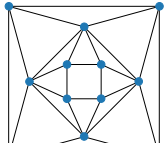
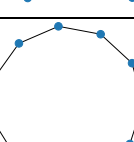
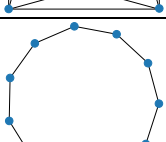





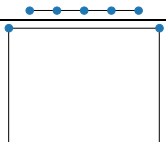
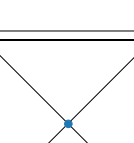
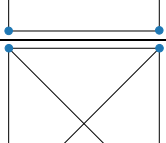
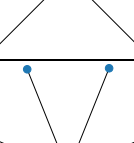
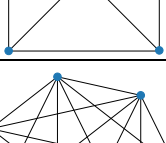
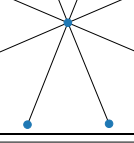
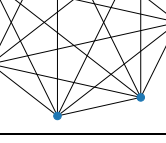
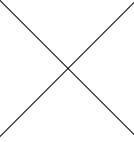
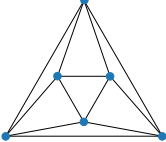
版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

表 2. 一些图 G 和其线图 $L(G)$; 参考 <https://mathworld.wolfram.com/LineGraph.html>

Graph G		Line graph $L(G)$	
claw graph $K_{1,3}$		triangle graph C_3	
complete bipartite graph $K_{2,3}$		prism graph Y_3	
cubical graph		cuboctahedral graph	
cycle graph C_n		cycle graph C_n	
path graph P_2		singleton graph K_1	
path graph P_n		path graph P_{n-1}	
square graph C_4		square graph C_4	
star graph S_5		tetrahedral graph K_4	
star graph S_n		complete graph K_{n-1}	
tetrahedral graph K_4		octahedral graph	
triangle graph C_3		triangle graph C_3	

常见图的关联矩阵

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

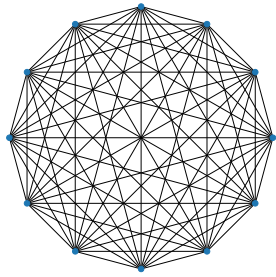
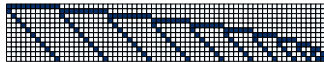
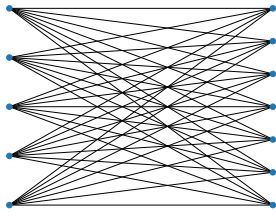
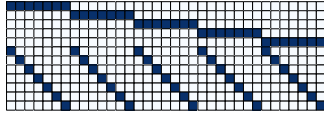
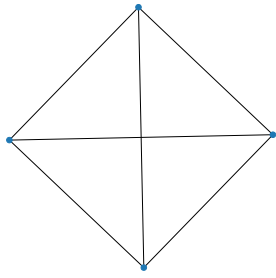
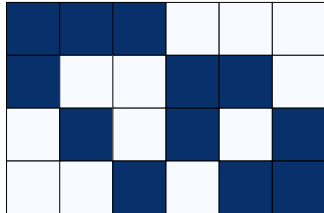
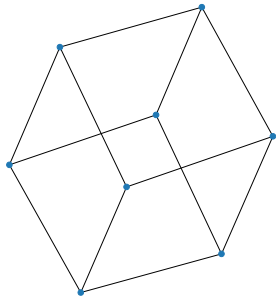
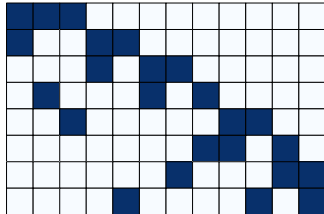
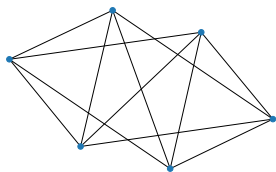
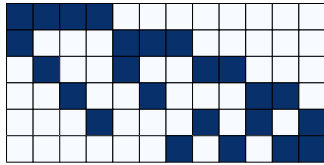
代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

表 3 总结常见图及其关联矩阵，请大家自行分析关联矩阵的特征，并对比前文相同图的邻接矩阵。Bk6_Ch21_04.ipynb 绘制表 3 中图和关联矩阵热图，请大家自行学习。

表 3. 常见图及其关联矩阵

常见图	图	关联矩阵
完全图		
完全二分图		
正四面体图		
正六面体图		
正八面体图		

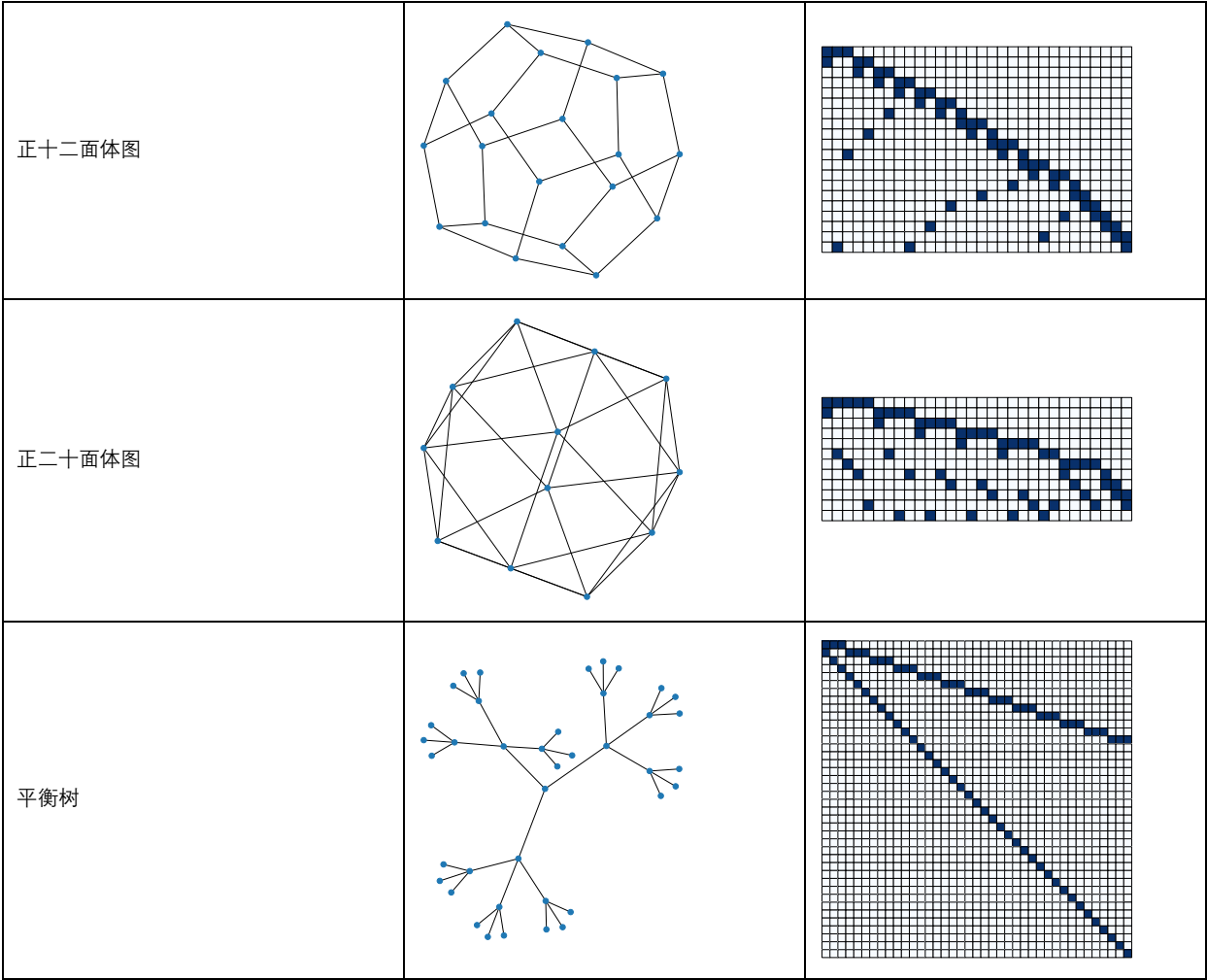
本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com



有向图

图 8 回顾前文介绍的有向图和邻接矩阵关系。

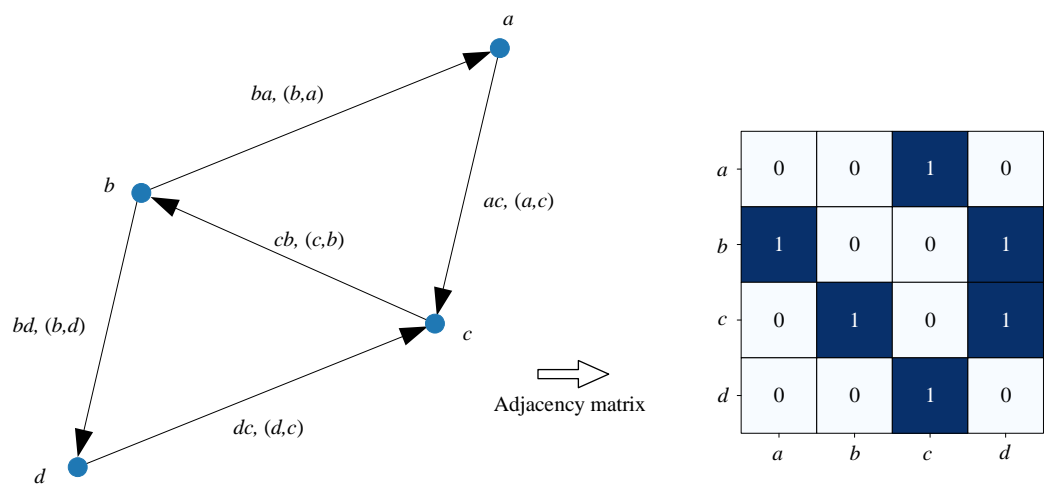


图 8. 从有向图到邻接矩阵热图

图 8 这幅有向图对应的关联矩阵为：

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。
版权归清华大学出版社所有，请勿商用，引用请注明出处。
代码及 PDF 文件下载：<https://github.com/Visualize-ML>
本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>
欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

$$C = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & -1 & 1 & 0 \\ 1 & 0 & 0 & -1 & 1 \\ 0 & 0 & 1 & 0 & -1 \end{bmatrix} \quad (5)$$

图9所示为从有向图到关联矩阵热图。

(5) 中关联矩阵列元素之和均为0，具体为

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (6)$$

这也不难理解，有向图的每一列代表一条有向边。不考虑自环、不考虑多图，有向边有入(+1)、有出(-1)。

(5) 中关联矩阵每行“+1”元素求和为节点的入度

$$\begin{bmatrix} 1 \\ 1 \\ 2 \\ 1 \end{bmatrix} \quad (7)$$

(5) 中关联矩阵每行“-1”元素求和取正为节点的出度

$$\begin{bmatrix} 1 \\ 2 \\ 1 \\ 1 \end{bmatrix} \quad (8)$$

图10逐个元素解释有向图和关联矩阵之间的关系，请大家自行分析四副子图。

大家可以在 Bk6_Ch21_05.ipynb 找到相关计算。

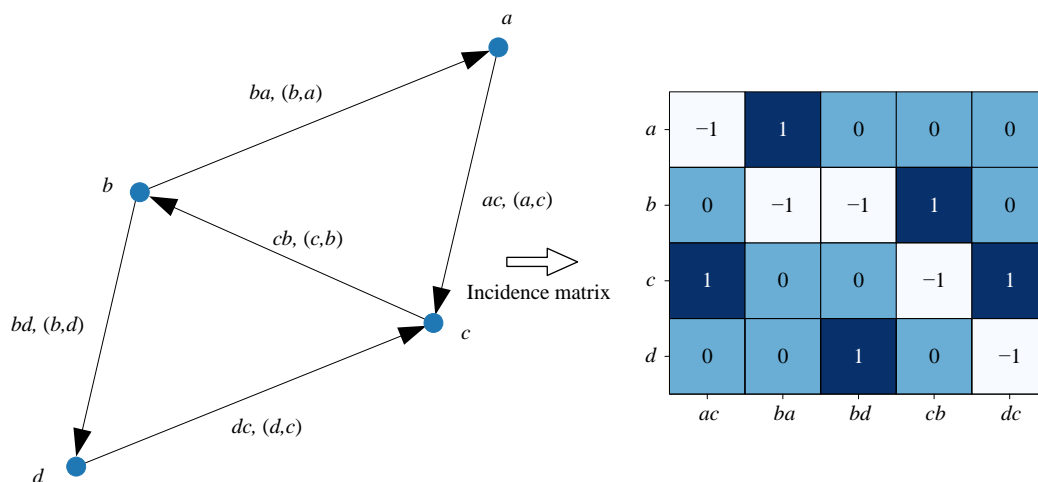


图 9. 从有向图到关联矩阵热图

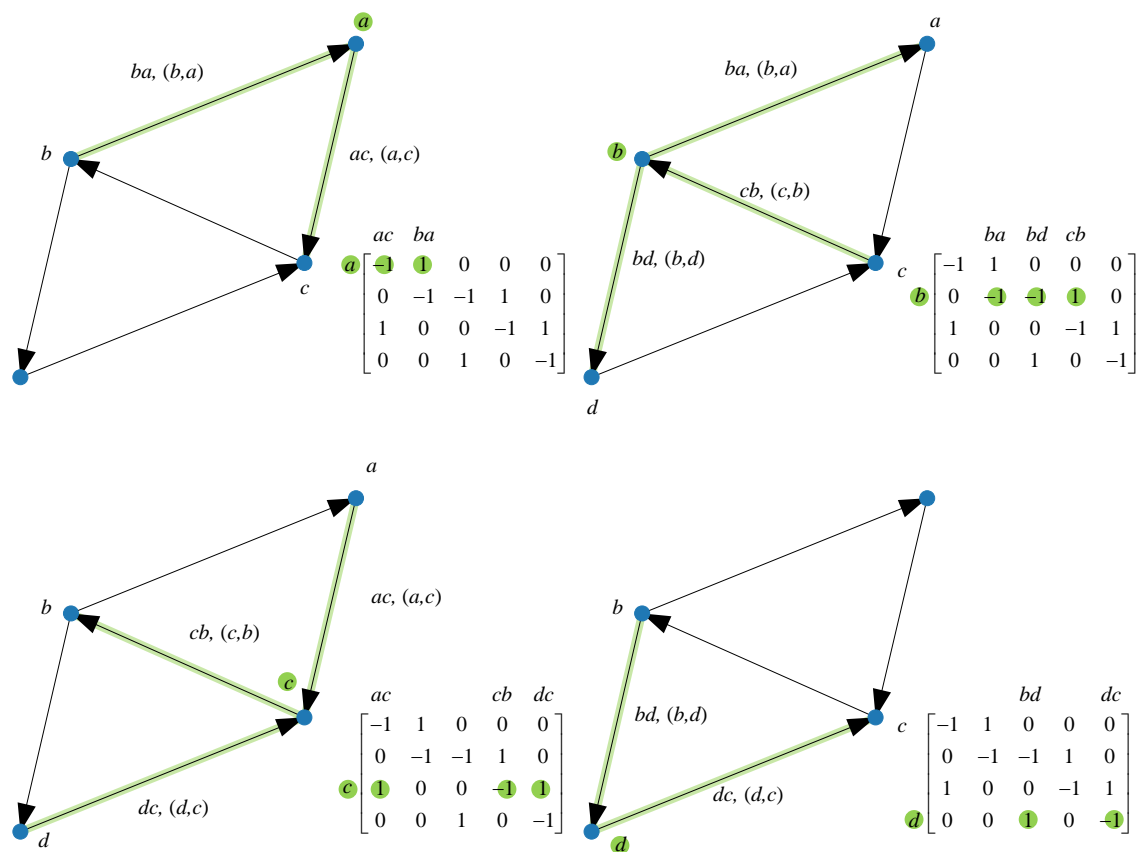


图 10. 逐个元素解释有向图和关联矩阵之间的关系

21.3 度矩阵

本书前文已经介绍过**度** (degree) 这个概念，本节将其扩展到**度矩阵** (degree matrix) 这个概念。

简单来说，度矩阵是一个与图的节点相关的矩阵，用于表示每个节点的度。对于一个图 G ，其度矩阵 D 是一个 $n \times n$ 的矩阵， n 表示图中节点的数量。

如果 G 中的节点 i 的度为 d_i ，那么度矩阵 D 的第 i 行第 i 列的元素为 d_i ；度矩阵 D 非主对角线上其他元素为 0。

本书前文介绍过，对于无向图，其邻接矩阵 A 沿列或行求和可以得到每个节点的度构成的向量。再将这个向量转换成对角矩阵，我们便得到度矩阵 D

$$D = \text{diag}(I^T A) \quad (9)$$

图 11 所示为无向图 G 的度矩阵。度矩阵的对角线元素表示每个节点的度，而非对角线元素均为零。显然，度矩阵 D 是对角方阵。

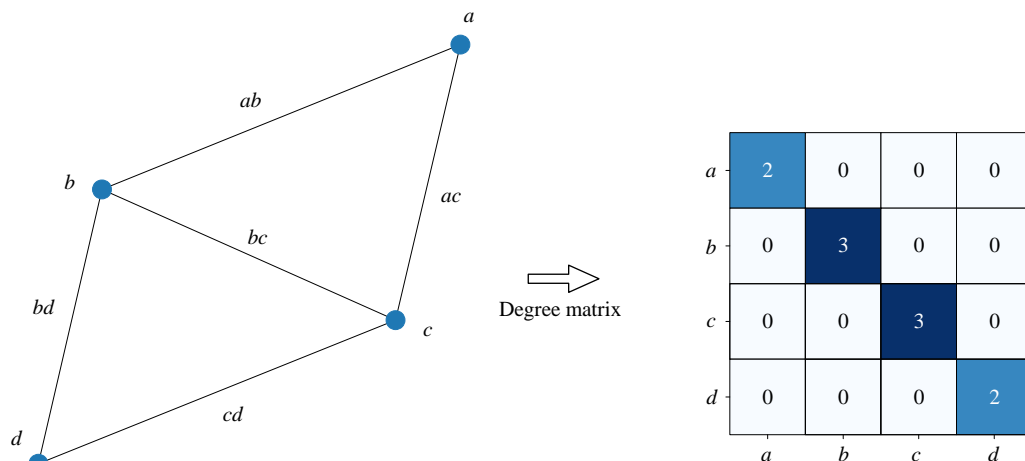


图 11. 无向图到度矩阵

Bk6_Ch21_05.ipynb 绘制图 11，还绘制图 12。图 12 是空手道俱乐部人员关系图对应的度矩阵热图。Bk6_Ch21_05.ipynb 这段代码很简单，请大家自行学习。

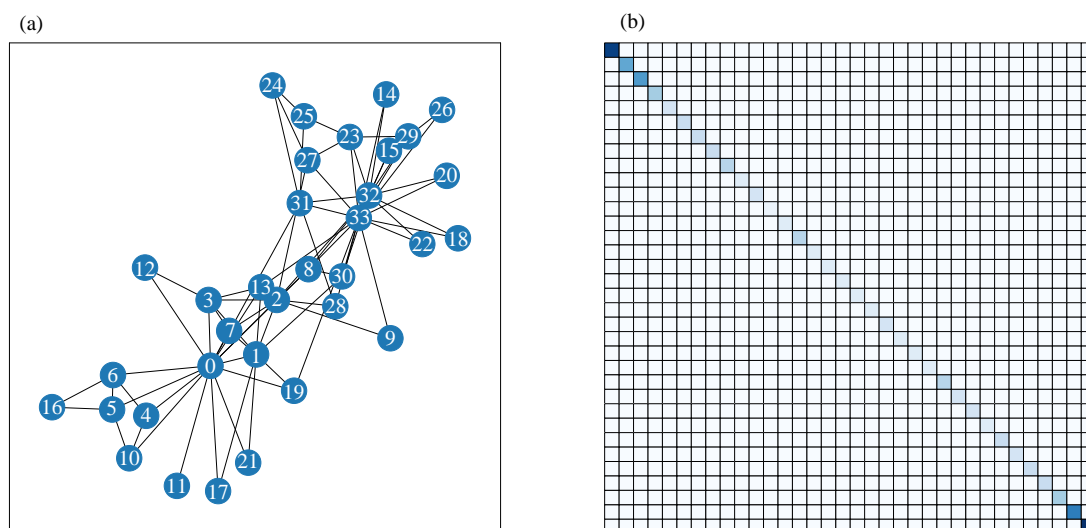


图 12. 空手道俱乐部人员关系图，以及对度矩阵热图

度矩阵可以用于分析网络的结构，识别中心节点，研究节点的重要性等。度矩阵是计算图的拉普拉斯矩阵的基础，这是下一节要介绍的内容。

对于有向图，我们可以分别得到它的**入度矩阵** (in-degree matrix) 和**出度矩阵** (out-degree matrix)。

Bk6_Ch21_06.ipynb 绘制图 13 两幅热图，代码很简单，请大家自行学习。

请大家复习本书前文介绍的**度分析** (degree analysis)。

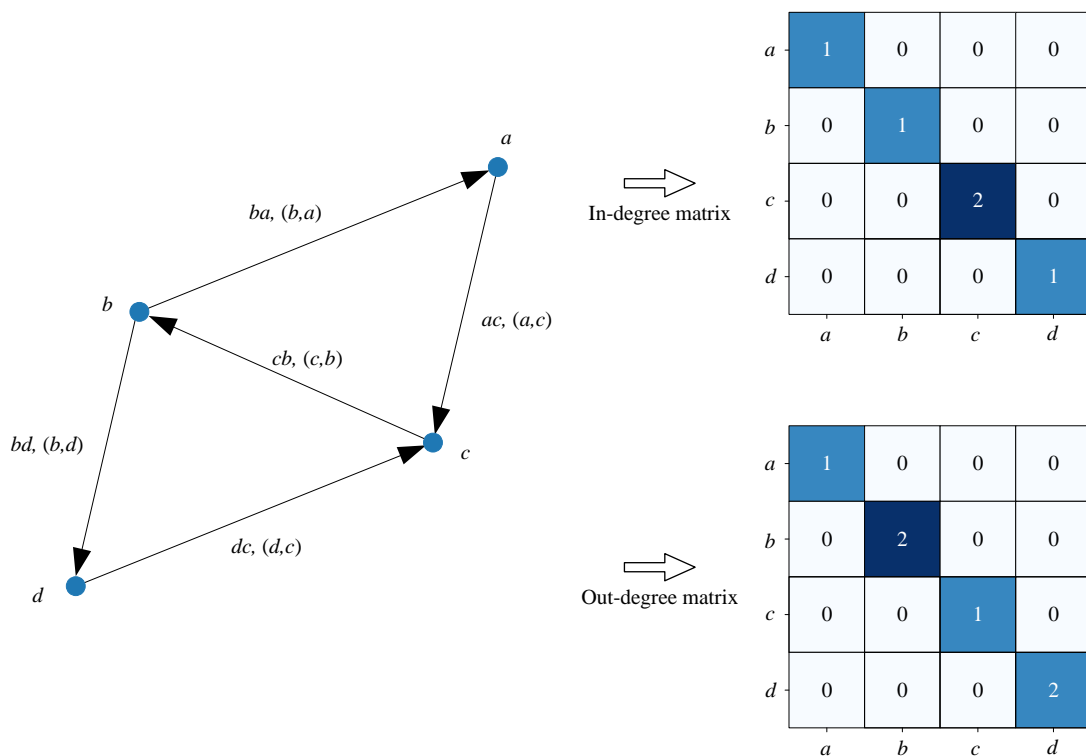


图 13. 从有向图到邻接矩阵热图

21.4 拉普拉斯矩阵

拉普拉斯矩阵 (Laplacian matrix) 是图论中的一个重要概念，通常用于表示图的结构和连接关系。在聚类问题中，拉普拉斯矩阵可以用来描述数据点之间的相似性，并通过对其进行**谱分解** (spectral decomposition) 来实现聚类。这是本书后续要介绍的一个话题。

对于无向图，拉普拉斯矩阵有几种不同的定义，其中最常见定义如下：

$$L = D - A \quad (10)$$

其中， D 是无向图的度矩阵， A 是无向图的邻接矩阵。

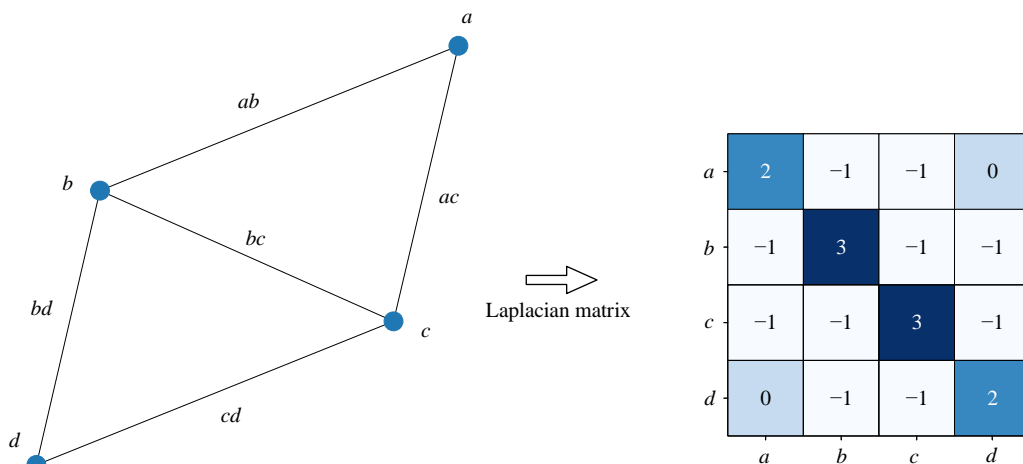


图 14. 无向图到拉普拉斯矩阵

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

$$L = D - A$$

2	-1	-1	0
-1	3	-1	-1
-1	-1	3	-1
0	-1	-1	2

2	0	0	0
0	3	0	0
0	0	3	0
0	0	0	2

0	1	1	0
1	0	1	1
1	1	0	1
0	1	1	0

图 15. 计算拉普拉斯矩阵

归一化拉普拉斯矩阵

对于无向图，**归一化拉普拉斯矩阵** (normalized Laplacian matrix) 定义如下：

$$L_n = D^{-1/2} (D - A) D^{-1/2} \quad (11)$$

也叫**归一化对称拉普拉斯矩阵** (normalized symmetric Laplacian matrix)。

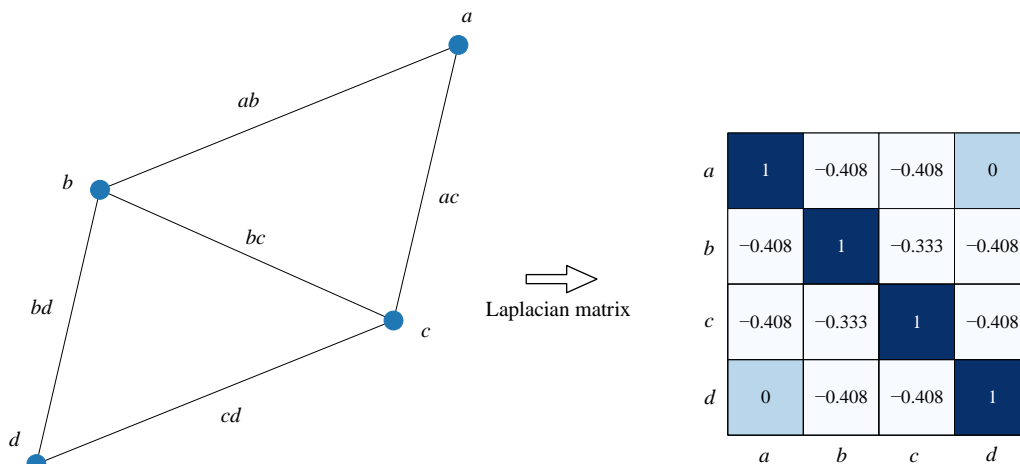


图 16. 无向图到归一化拉普拉斯矩阵

Bk6_Ch21_07.ipynb 绘制图 14 和图 16，并验证两个拉普拉斯矩阵计算过程，下面聊聊其中关键语句。

- a** 用 `networkx.laplacian_matrix()` 获取无向图的拉普拉斯矩阵。
- b** 用 `networkx.adjacency_matrix()` 计算无向图的邻接矩阵。
- c** 计算无向图的度矩阵。
- d** 验证拉普拉斯矩阵。
- e** 用 `networkx.normalized_laplacian_matrix()` 计算无向图的归一化拉普拉斯矩阵。
- f** 验证归一化拉普拉斯矩阵。

表 4 总结常见图的归一化拉普拉斯矩阵，请大家自己寻找规律；Bk6_Ch21_08.ipynb 绘制表 4 图和热图，请大家自行学习。

```

import matplotlib.pyplot as plt
import networkx as nx
import numpy as np
import seaborn as sns

G = nx.Graph()
# 创建无向图的实例

G.add_nodes_from(['a', 'b', 'c', 'd'])
# 添加多个顶点

G.add_edges_from([( 'a', 'b' ), ( 'b', 'c' ),
                   ( 'b', 'd' ), ( 'c', 'd' ),
                   ( 'c', 'a' )])
# 增加一组边

# 计算拉普拉斯矩阵
a L = nx.laplacian_matrix(G).toarray()

b A = nx.adjacency_matrix(G).todense()
# 邻接矩阵

c D = A.sum(axis = 0)
D = np.diag(D)
# 度矩阵

# 验证拉普拉斯矩阵
d D - A

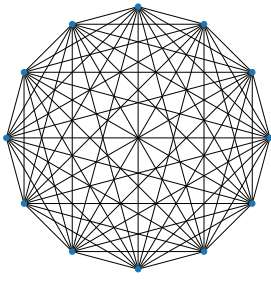
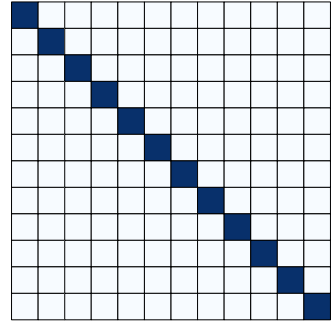
# 归一化（对称）拉普拉斯矩阵
e L_N = nx.normalized_laplacian_matrix(G).todense()

# 验证归一化拉普拉斯矩阵
f D_sqrt_inv = np.diag(1/np.sqrt(A.sum(axis = 0)))
D_sqrt_inv @ L @ D_sqrt_inv

```

代码 3. 拉普拉斯矩阵 | Bk6_Ch21_06.ipynb

表 4. 常见图及其归一化拉普拉斯矩阵

常见图	图	归一化拉普拉斯矩阵
完全图		

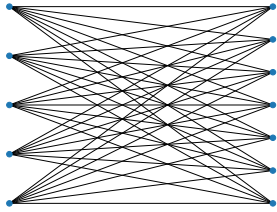
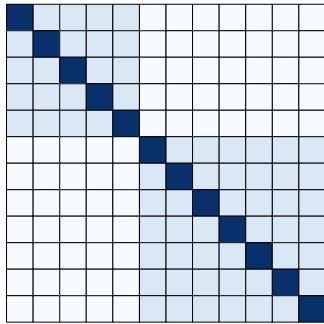
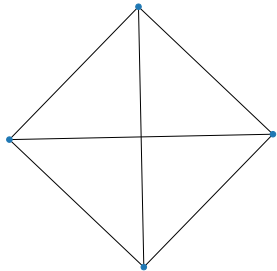
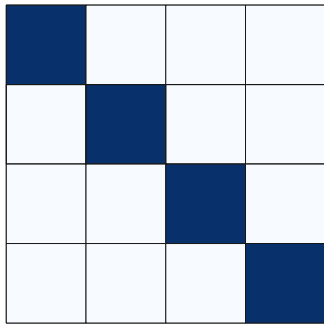
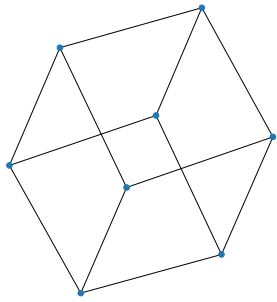
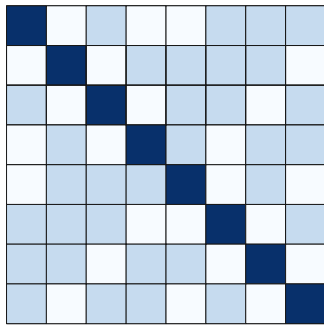
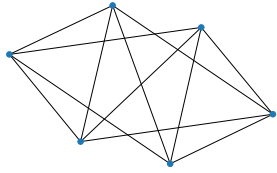
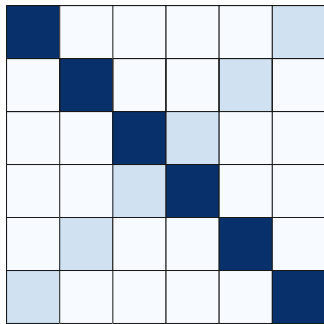
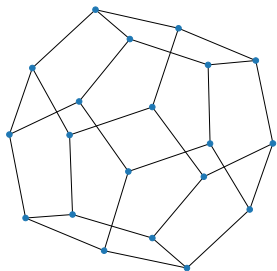
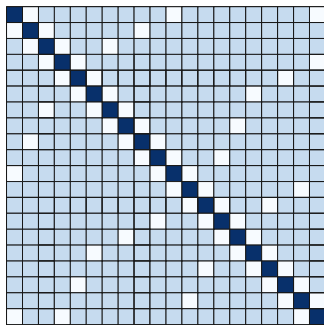
本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

完全二分图		
正四面体图		
正六面体图		
正八面体图		
正十二面体图		

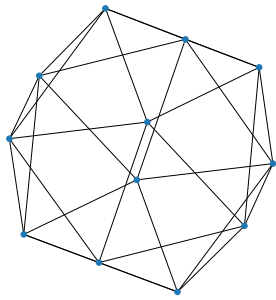
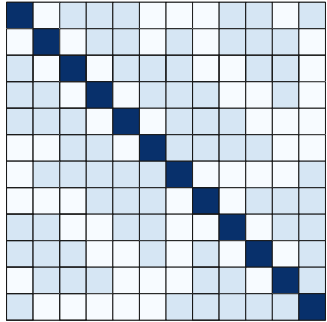
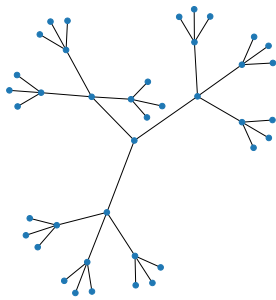
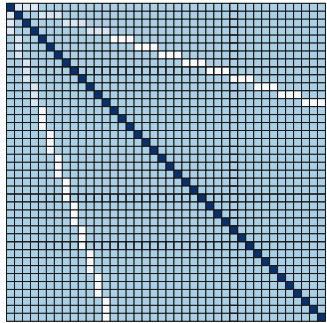
本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

正二十面体图		
平衡树		

拉普拉斯矩阵谱分解

拉普拉斯矩阵的谱分解将图的结构信息编码到其特征值和特征向量中。通过对拉普拉斯矩阵进行谱分解，可以得到一组特征向量，这些特征向量对应于图的不同谱分量。这些特征向量可以用于聚类，因为相似的节点在谱空间中通常会被映射到相似的位置。

谱排序 (spectral ordering) 是一种基于图的谱性质的节点排序方法。谱排序的基本思想是，通过对图的拉普拉斯矩阵的特征向量进行排序，得到的排序顺序将具有一定的图结构信息。通常，这种排序方法可以用于提取图的特征，发现图中的模式或社区结构。

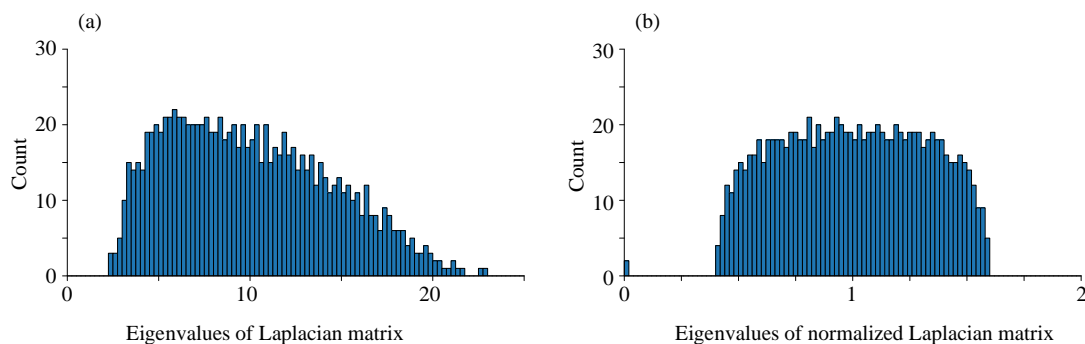


图 17. 拉普拉斯矩阵谱分析结果

- a** 用 `networkx.gnm_random_graph()` 创建图。
- b** 用 `networkx.laplacian_spectrum(G)` 完成无向图 `G` 的拉普拉斯矩阵的谱分析。
- c** 被注释掉的这两句用来验证拉普拉斯矩阵谱分析结果。先用 `networkx.laplacian_matrix()` 计算拉普拉斯矩阵，然后再用 `numpy.linalg.eigvals()` 计算拉普拉斯矩阵特征值。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

- d 用 `matplotlib.pyplot.hist()` 绘制拉普拉斯特征值直方图。
- e 用 `networkx.normalized_laplacian_spectrum(G)` 完成无向图 G 的归一化拉普拉斯矩阵的谱分析。
- f 同样也是用来验证归一化拉普拉斯矩阵谱分析结果。
- g 也是用 `matplotlib.pyplot.hist()` 绘制归一化拉普拉斯特征值直方图。

```

import matplotlib.pyplot as plt
import networkx as nx
import numpy.linalg

n = 1000 # 1000 nodes
m = 5000 # 5000 edges
a G = nx.gnm_random_graph(n, m, seed=8)
# 创建图

# 拉普拉斯矩阵谱分解
b eig_values_L = nx.laplacian_spectrum(G)

# 验证
c # L = nx.laplacian_matrix(G)
# numpy.linalg.eigvals(L.toarray())
print("Largest eigenvalue:", max(eig_values_L))
print("Smallest eigenvalue:", min(eig_values_L))

# 可视化
d fig, ax = plt.subplots(figsize = (6,3))
ax.hist(eig_values_L, bins=100,
        ec = 'k', range = [0,25])
ax.set_ylabel("Count")
ax.set_xlabel("Eigenvalues of Laplacian matrix")
ax.set_xlim(0,25)
ax.set_ylim(0,30)
plt.savefig('拉普拉斯矩阵谱.svg')

# 归一化拉普拉斯矩阵谱分解
e eig_values_L_N = nx.normalized_laplacian_spectrum(G)

f # L_N = nx.normalized_laplacian_matrix(G)
# numpy.linalg.eigvals(L_N.toarray())

print("Largest eigenvalue:", max(eig_values_L_N))
print("Smallest eigenvalue:", min(eig_values_L_N))

# 可视化
g fig, ax = plt.subplots(figsize = (6,3))
ax.hist(eig_values_L_N, bins=100,
        ec = 'k', range = [0,2])
ax.set_ylabel("Count")
ax.set_xlabel("Eigenvalues of normalized Laplacian matrix")
ax.set_xlim(0,2)
ax.set_ylim(0,20)
plt.savefig('归一化拉普拉斯矩阵谱.svg')

```

代码 4. 拉普拉斯矩阵的谱分析 | Bk6_Ch21_09.ipynb

下面我们以空手道俱乐部数据为例简单介绍如何用谱分解拉普拉斯矩阵完成聚类。

图 18 (a) 展示空手道俱乐部人员关系图；图 18 (b) 用热图可视化其拉普拉斯矩阵。

然后利用特征值分解（准确来说是谱分解，因为拉普拉斯矩阵为对称矩阵）拉普拉斯矩阵。图 19 (a) 所示为特征向量构成的矩阵，特征向量从左到右根据特征值从小到大排序。图 19 (b) 对角方阵的对角线元素为特征值。图 19 (c) 为特征值从小到大的线图。

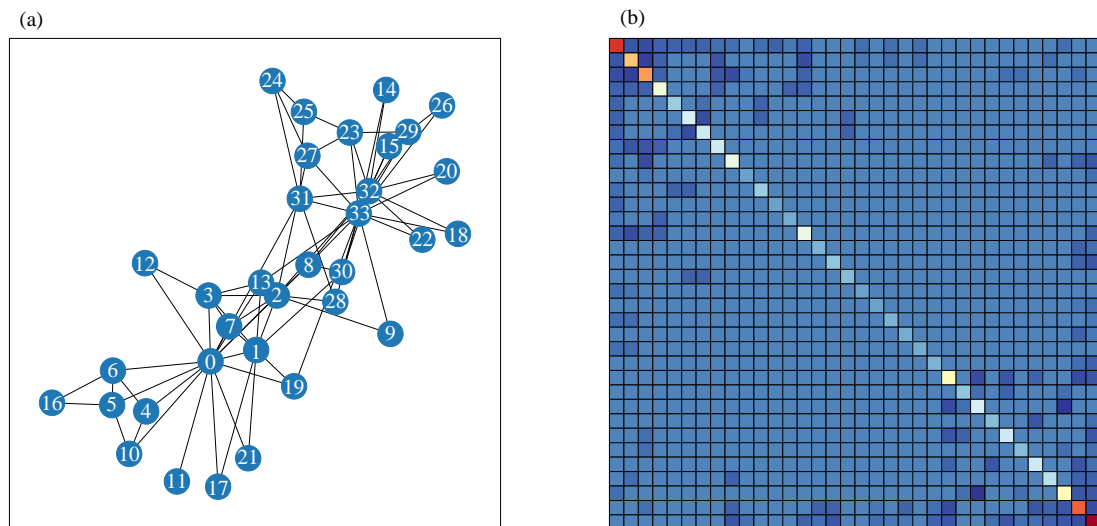


图 18. 空手道俱乐部人员关系图，以及对应拉普拉斯矩阵热图

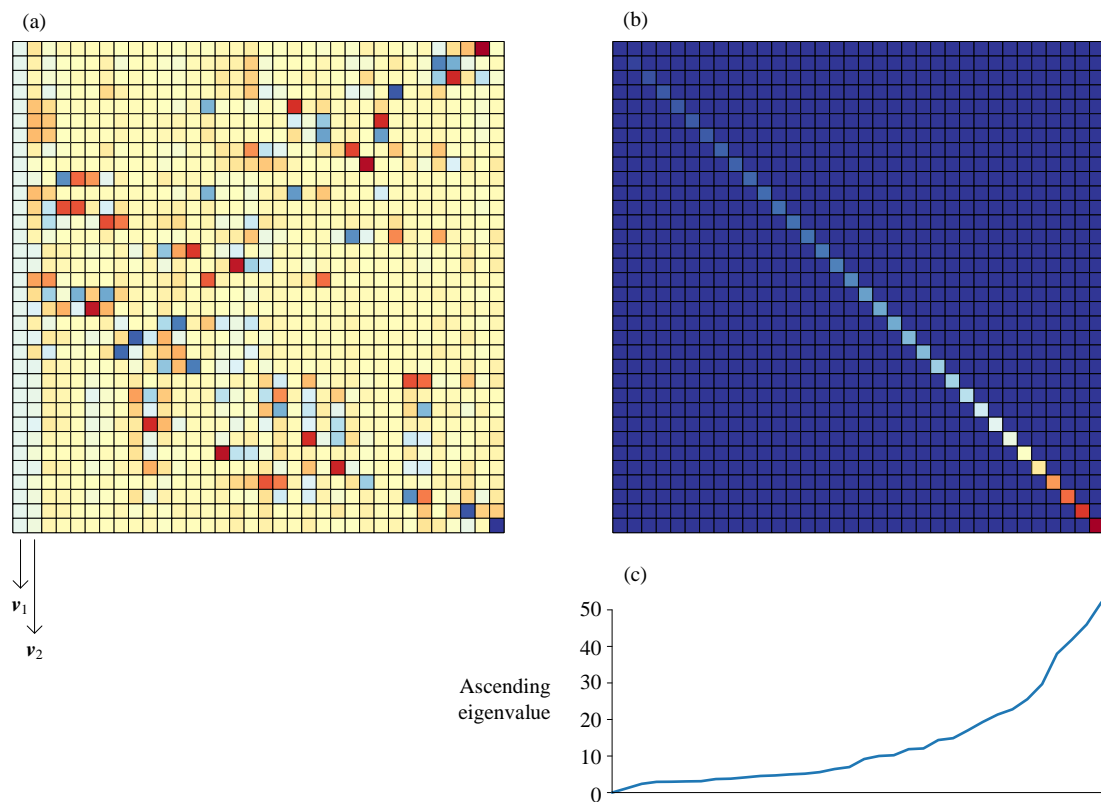


图 19. 拉普拉斯矩阵谱分解结果

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

图 20 所示为前两个特征向量对应的散点图；很容易发现，沿着 $y = 0$ 切一刀，节点就可以分成两簇，对应结果如图 21 所示。

本书后续在介绍**谱聚类** (spectral clustering) 时，还会用到拉普拉斯矩阵的谱分解。

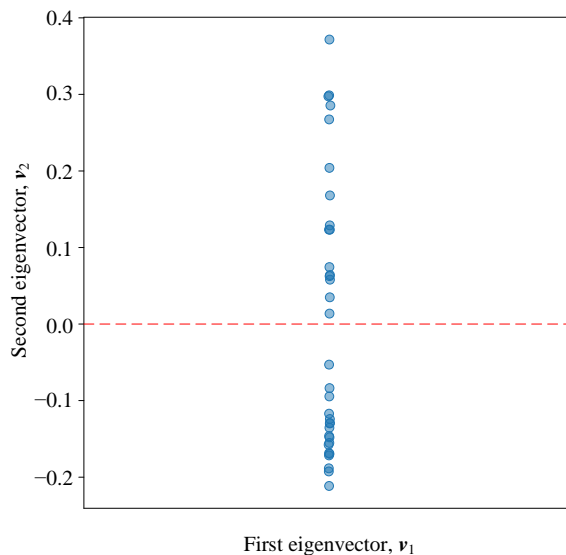


图 20. 前两个特征向量散点图

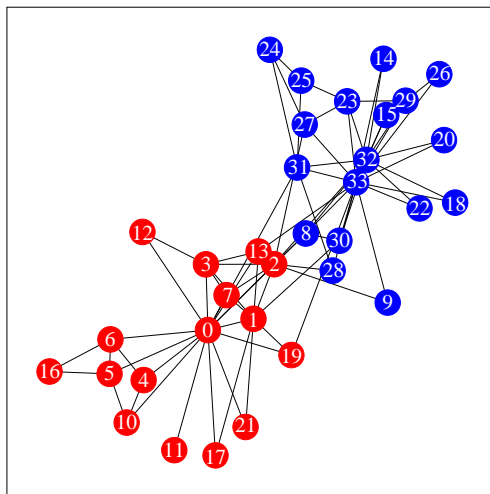


图 21. 根据前两个特征向量绘制的散点图完成聚类

代码 5 完成上述运算，下面聊聊其中关键语句。

- a 用 `networkx.karate_club_graph()` 加载空手道俱乐部数据。
- b 用 `networkx.laplacian_matrix(G)` 计算图 G 的拉普拉斯矩阵。
- c 用 `numpy.linalg.eig()` 对拉普拉斯矩阵特征值分解（谱分解）。
- d 根据特征值从小到大排序特征向量。
- e 对于第 2 特征向量，以 0 为界，分别用不同颜色标记节点。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com


```

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import networkx as nx
import seaborn as sns

a G = nx.karate_club_graph()
# 空手道俱乐部图
pos = nx.spring_layout(G, seed=2)

b L = nx.laplacian_matrix(G).todense()
# 拉普拉斯矩阵

c lambdas, V = np.linalg.eig(L)
# 特征值分解

# 按特征值从小到大排列
d lambdas_sorted = np.sort(lambdas)
V_sorted = V[:, lambdas.argsort()]

# 聚类标签
e colors = [ "r" for i in range(0,34)]
for i in range(0,34):
    if (V_sorted[i,1] < 0):
        colors[i] = "b"

plt.figure(figsize = (6,6))
nx.draw_networkx(G, pos,
                 # with_labels = False,
                 node_color=colors)
plt.savefig('图节点聚类.svg')

```

代码 5. 谱分解拉普拉斯矩阵用来聚类 | Bk6_Ch21_10.ipynb

总结来说，关联矩阵是一种紧凑的方式来表示图结构、分析图的性质，尤其是在计算机算法和图论算法中。在网络分析中，关联矩阵常用于表示社交网络、交通网络等，并通过矩阵运算来研究网络的性质。关联矩阵可以用于建模和求解一些优化问题。

谱排序在一些图分析和图算法中有应用，例如在图划分、社区检测和图可视化等领域。然而，具体的谱排序方法可能因应用场景而异，因此在具体使用时需要注意选择适当的特征向量和排序策略。

有向图的拉普拉斯矩阵，请大家参考：

https://networkx.org/documentation/stable/reference/generated/networkx.linalg.laplacianmatrix.directed_laplacian_matrix.html