

# 18

## From Graphs to Matrices

# 从图到矩阵

图就是矩阵，矩阵就是图



如果我在沉睡一千年后醒来，我的第一个问题是：黎曼猜想被证明了吗？

***If I were to awaken after having slept for a thousand years, my first question would be: has the Riemann Hypothesis been proven?***

—— 大卫·希尔伯特 (David Hilbert) | 德国数学家 | 1862 ~ 1943



- networkx.circular\_layout() 节点圆周布局
- networkx.DiGraph() 创建有向图的类，用于表示节点和有向边的关系以进行图论分析
- networkx.draw\_networkx() 用于绘制图的节点和边，可根据指定的布局将图可视化呈现在平面上
- networkx.get\_edge\_attributes() 获取图中边的特定属性的字典
- networkx.get\_node\_attributes() 获取图中节点的特定属性的字典
- networkx.Graph() 创建无向图的类，用于表示节点和边的关系以进行图论分析
- networkx.relabel\_nodes() 对节点重命名
- networkx.to\_numpy\_matrix() 用于将图表示转换为 NumPy 矩阵，方便在数值计算和线性代数操作中使用
- numpy.linalg.norm() 计算范数

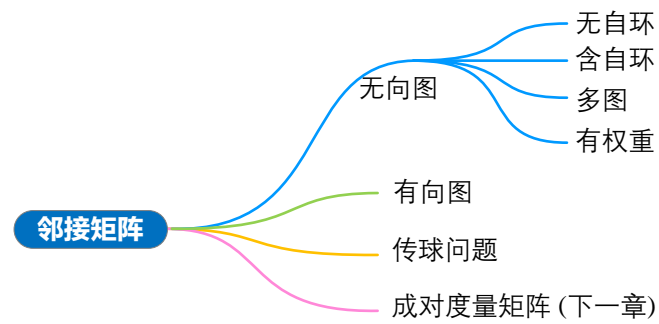
本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

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](mailto:jiang.visualize.ml@gmail.com)



## 18.1 无向图到邻接矩阵

请大家记住一句话，矩阵就是图，图就是矩阵。

大家可能好奇，图怎么和矩阵扯上关系？本章就试着回答这个问题。

在图论中，**邻接矩阵** (adjacency matrix) 是一种用于表示图的矩阵。

对于无向图，邻接矩阵是一个对称矩阵，其中行和列的数量等于图中的节点数量，矩阵的元素表示节点之间是否存在边。

不考虑权重的条件下，对于一个无向图  $G$ ，其邻接矩阵  $A$  的定义如下：

如果节点  $i$  和节点  $j$  之间存在边，则  $a_{i,j}$  和  $a_{j,i}$  的值为 1；

如果节点  $i$  和节点  $j$  之间不存在边，则  $a_{i,j}$  和  $a_{j,i}$  的值为 0。

根据无向图邻接矩阵定义，上述矩阵  $A$  一定是**对称矩阵** (symmetric matrix)。

邻接矩阵的优势之一是它提供了一种紧凑的方式来表示图中的连接关系，并且对于某些图算法，邻接矩阵的形式更易于处理。

下面，我们通过实例具体讨论不同类型无向图对应的邻接矩阵。

### 无自环

首先我们先看一下如何用邻接矩阵来表达无自环无向图。

相信大家已经很熟悉图 1 中这幅无自环无向图，我们在本书前文经常用到这幅图做例子。图 1 还给出了这幅图对应的邻接矩阵  $A$

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

由于无向图的邻接矩阵为对称矩阵，因此我们仅仅需要存储上三角或下三角部分数据。

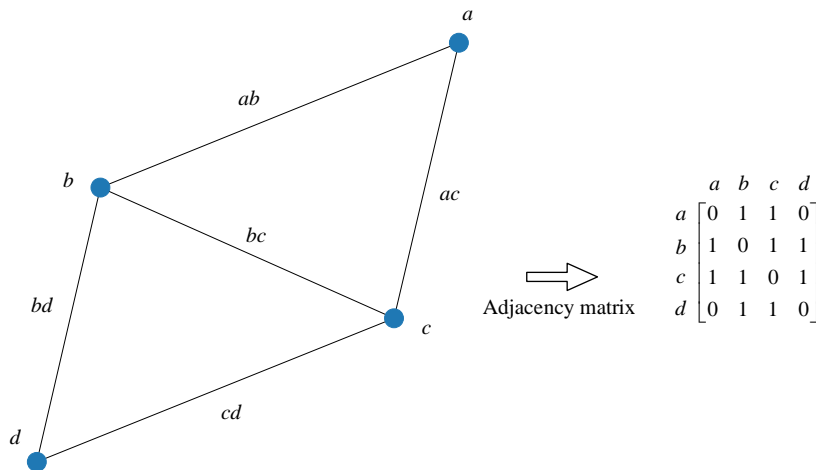


图 1. 从无向图到邻接矩阵，无自环

由于图 1 无向图有 4 个节点 ( $a$ 、 $b$ 、 $c$ 、 $d$ )。因此，邻接矩阵  $A$  的形状为  $4 \times 4$ 。邻接矩阵  $A$  的 4 行从上到下分别代表 4 个节点—— $a$ 、 $b$ 、 $c$ 、 $d$ 。同样，邻接矩阵  $A$  的 4 列从左到右也分别代表这 4 个节点。

图 2 逐个元素解释无自环无向图和邻接矩阵之间的关系。

举个例子，由于节点  $a$ 、 $b$  之间存在一条无向边  $ab$ ，因此邻接矩阵  $A$  中  $a_{1,2}$  和  $a_{2,1}$  元素都为 1。显然， $a_{1,2}$  和  $a_{2,1}$  关于主对角线对称，这也解释了为什么无向图的邻接矩阵  $A$  为对称矩阵。

看到对称矩阵，大家是否眼前一亮？是否联想到了特征值分解和谱分解？矩阵分解和对称矩阵又能碰撞出怎样的火花？这是本书后文要介绍的内容。

由于图 8 无向图不含自环，矩阵  $A$  对角线元素为 0。

请大家自己分析图 2 剩余子图。

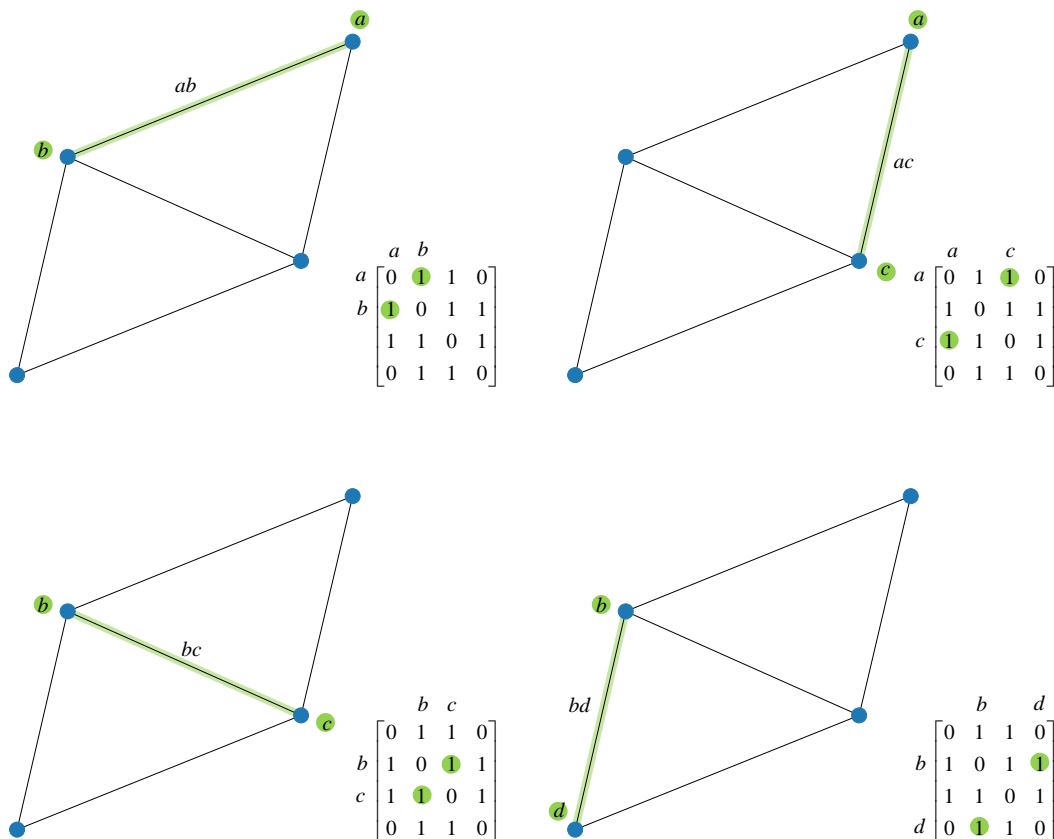


图 2. 逐个元素解释无自环无向图和邻接矩阵之间的关系

此外，矩阵  $A$  沿列求和可以得到每个节点的度。比如，(1) 沿列求和：

$$I^T A = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 2 & 3 & 3 & 2 \end{bmatrix} \quad (2)$$

同样，对于无向图，矩阵  $A$  沿行求和也可以得到每个节点的度

$$A I = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 3 \\ 3 \\ 2 \end{bmatrix} \quad (3)$$

(2) 和 (3) 结果互为转置。

表 1 展示 4 个构造的几种 (不含自环、不加权) 无向图和它们的邻接矩阵。建议大家做两个练习。第一个练习，遮住邻接矩阵，将无向图写成邻接矩阵；第二个练习，遮住无向图，根据邻接矩阵绘制无向图。

这两个练习也告诉我们，我们可以将无向图和邻接矩阵相互转换。而 NetworkX 就有相应工具完成这种转换。

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。

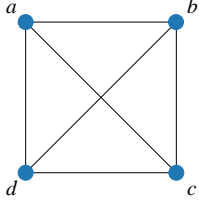
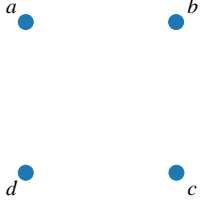
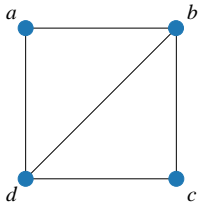
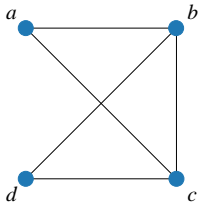
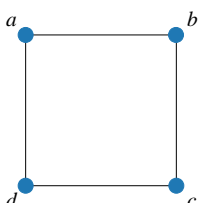
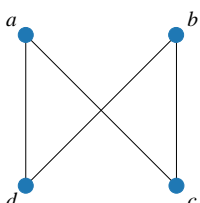
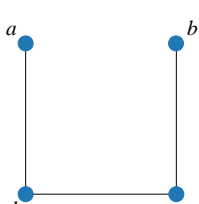
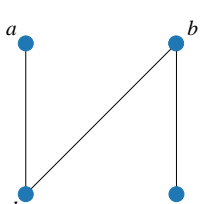
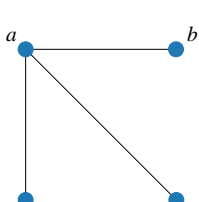
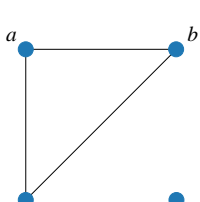
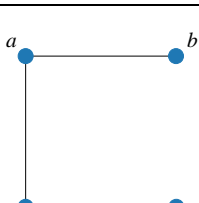
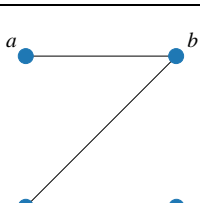
版权归清华大学出版社所有，请勿商用，引用请注明出处。

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](mailto:jiang.visualize.ml@gmail.com)

表 1.4 个节点构造的几种无向图及邻接矩阵，不含自环，不加权

无向图	邻接矩阵	无向图	邻接矩阵
	$\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$		$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$
	$\begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$		$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$
	$\begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$		$\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}$
	$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$		$\begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}$
	$\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$		$\begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}$
	$\begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$		$\begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$

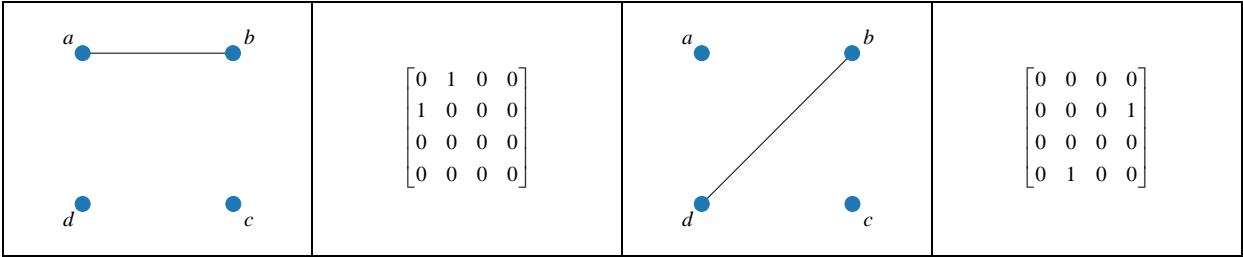
本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

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](mailto:jiang.visualize.ml@gmail.com)



代码 1 利用 NetworkX 工具将无向图转化为邻接矩阵。请大家注意 <sup>a</sup> 中 `to_numpy_matrix()` 方法。请大家按照代码 1 思路完成表 1 中几个无向图到邻接矩阵的转化。

```
import matplotlib.pyplot as plt
import networkx as nx

a undirected_G = nx.Graph()
# 创建无向图的实例

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

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

d adjacency_matrix = nx.to_numpy_matrix(undirected_G)
# 将无向图转换为邻接矩阵
```

代码 1. 将无向图转换为邻接矩阵 | Bk6\_Ch18\_01.ipynb

图 3 (a) 所示为空手道俱乐部人员关系无向图；图 3 (b) 所示为邻接矩阵热图。

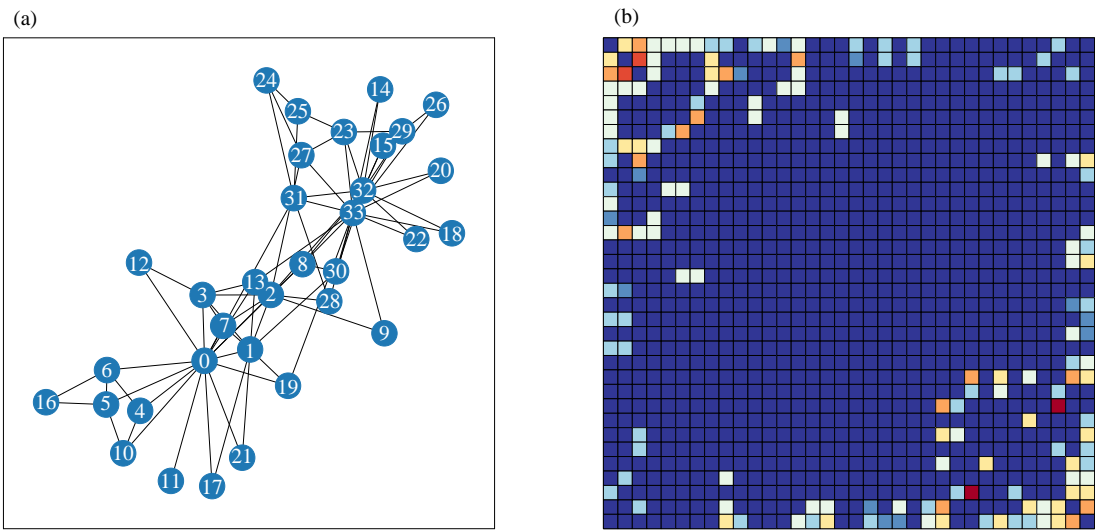


图 3. 空手道俱乐部人员关系图，以及对应邻接矩阵热图

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](mailto:jiang.visualize.ml@gmail.com)

代码 2 绘制图 3，代码相对简单，下面仅仅聊聊如下几句。


- a 用 `networkx.karate_club_graph()` 加载空手道俱乐部数据。
- b 用 `networkx.adjacency_matrix()` 计算空手道俱乐部图的邻接矩阵。
- c 用 `seaborn.heatmap()` 以热图的形式呈现邻接矩阵，颜色映射采用 `'RdYlBu_r'`。

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

plt.figure(figsize = (6,6))
nx.draw_networkx(G_karate,
                  pos = pos)
plt.savefig('空手道俱乐部图.svg')

b A_karate = nx.adjacency_matrix(G_karate).todense()
# 邻接矩阵

c sns.heatmap(A_karate, cmap = 'RdYlBu_r',
              square = True,
              xticklabels = [], yticklabels = [])
plt.savefig('A邻接矩阵.svg')
```

代码 2. 将空手道俱乐部无向图转换为邻接矩阵 |  Bk6\_Ch18\_01.ipynb

代码 3 将邻接矩阵转化为无向图。注意，c 是一个字典生成式，它用于创建一个将图中节点索引映射到小写英文字母的字典。`ord('a') + i` 计算出对应的 ASCII 值，然后 `chr()` 函数将这个 ASCII 值转换为对应的字符，即小写英文字母。

在绘制图时使用 `labels=node_labels` 参数，图中的节点将被标记为小写英文字母而不是默认的数字索引。这有助于提高图的可读性。

```
import numpy as np
import matplotlib.pyplot as plt
import networkx as nx

a adjacency_matrix = np.array([[0, 1, 1, 0],
                               [1, 0, 1, 1],
                               [1, 1, 0, 1],
                               [0, 1, 1, 0]])

# 定义邻接矩阵

b G = nx.Graph(adjacency_matrix, nodetype=int)
# 用邻接矩阵创建无向图

c node_labels = {i: chr(ord('a') + i) for i in range(len(G.nodes))}
# 创建字典，可视化时用作节点标签
# {0: 'a', 1: 'b', 2: 'c', 3: 'd'}

# 可视化
plt.figure(figsize = (6,6))
d nx.draw_networkx(G, with_labels=True, labels=node_labels)
```


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](mailto:jiang.visualize.ml@gmail.com)

代码 3. 将邻接矩阵转换为无向图 |  Bk6\_Ch18\_02.ipynb

## 含自环

图 4 中节点  $a$  增加自环后，图的邻接矩阵就变成了

$$A = \begin{bmatrix} 2 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \quad (4)$$

简单来说，如果节点有自我连接产生的自环，则在矩阵的主对角线上会有非零的值；如果没有自环，则主对角线上全部是 0。

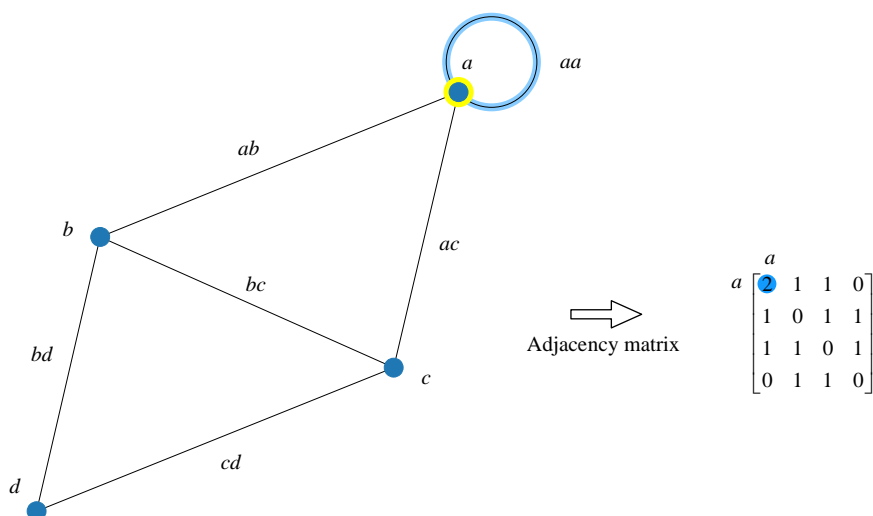


图 4. 节点  $a$  增加自环，转换成邻接矩阵

## 多图

本书前文提过，多图允许在同一对节点之间存在多条边。

图 5 的多图对应的邻接矩阵为

$$A = \begin{bmatrix} 0 & 2 & 2 & 1 \\ 2 & 0 & 0 & 1 \\ 2 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \quad (5)$$

很明显节点  $a$ 、 $b$  之间的边数为 2，节点  $a$ 、 $c$  之间的边数也是 2。



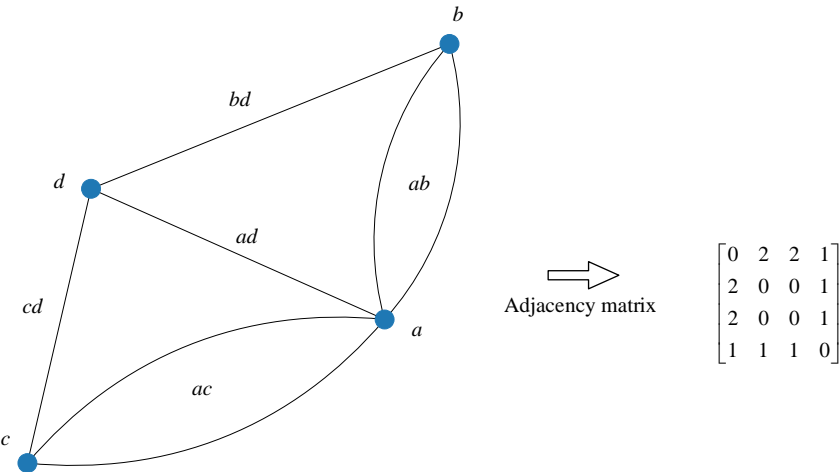


图 5. 多图

补图

简单图  $G$  补图  $\bar{G}$  的邻接矩阵  $\bar{A}$  可以通过下式求得

$$\bar{A} = J - I - A \tag{6}$$

其中， $A$  是图  $G$  的邻接矩阵， $J$  是全 1 方阵， $I$  是单位矩阵。  
也就是说，

$$\bar{A} + A = J - I \tag{7}$$

结果  $J - I$  为方阵，主对角线元素为 0，其余元素均为 1。而 (7) 正是和图  $G$  节点数相同的完全图的邻接矩阵。

图 6 给出的示例展示的就是图、补图、完全图三者的图和邻接矩阵之间的关系。

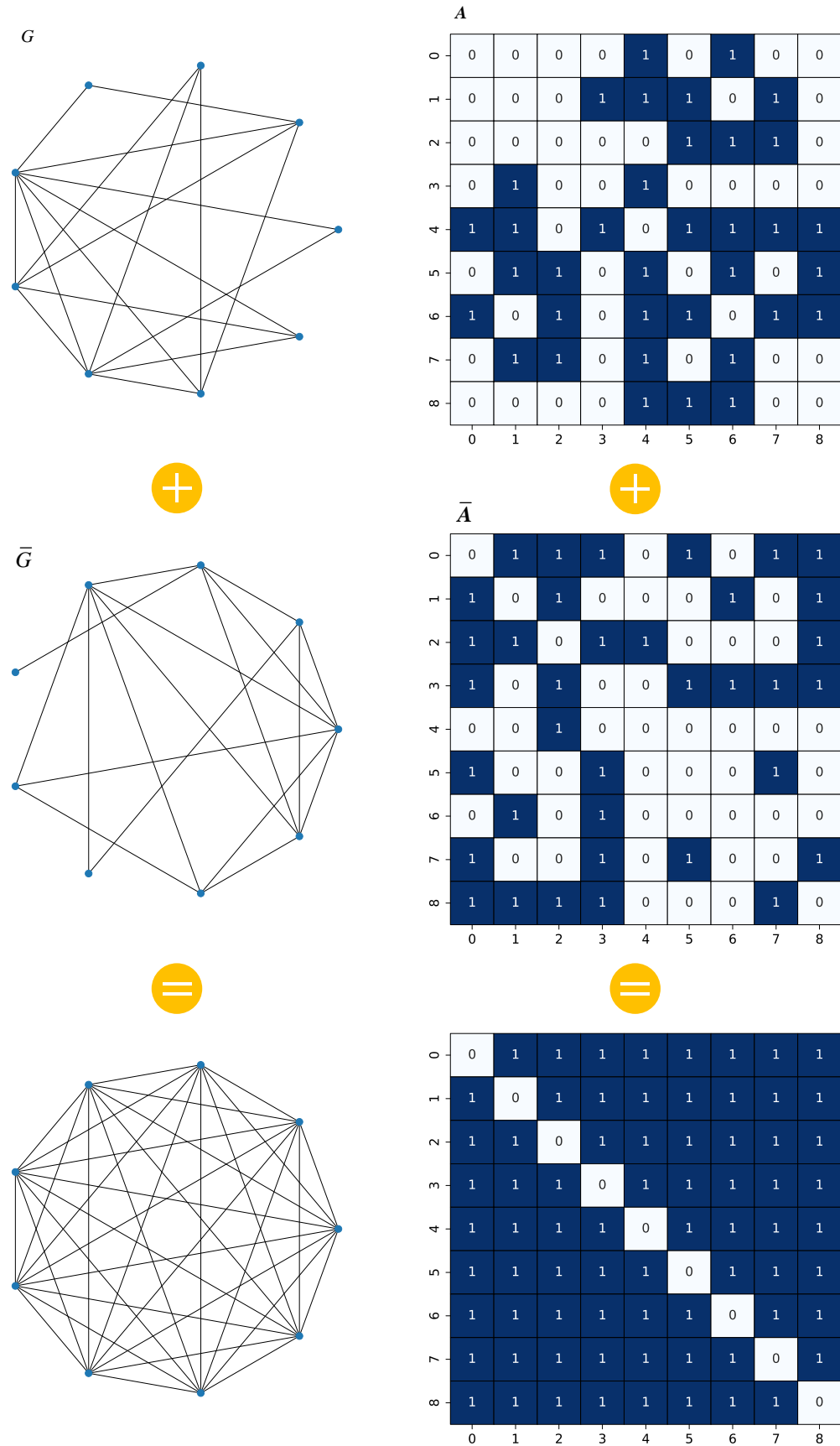


图 6. 图、补图、完全图三者的邻接矩阵

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](mailto:jiang.visualize.ml@gmail.com)

代码 4 绘制图 6，下面聊聊其中关键语句。

- a 自定可视化函数，绘制一行、两列子图布局图像；左子图为图，右子图为邻接矩阵热图。
- b 在左子图用 `networkx.draw_networkx()` 绘制图。参数 `ax` 指定子图的轴。
- c 用 `networkx.adjacency_matrix()` 计算图的邻接矩阵。
- d 在右子图用 `seaborn.heatmap()` 绘制邻接矩阵热图。
- e 用 `networkx.complete_graph()` 创建 9 个节点的完全图。
- f 用 `copy()` 方法获得图的副本，非视图。
- g 用 `remove_edges_from()` 方法随机删去 18 条边创建子图 G。
- h 用 `networkx.complement()` 创建子图 G 的补图。

```

import networkx as nx
import matplotlib.pyplot as plt
import random
import seaborn as sns

def visualize(G, fig_title):
    fig, axs = plt.subplots(nrows = 1, ncols = 2,
                           figsize = (12,6))
    pos = nx.circular_layout(G)
    # 左子图
    nx.draw_networkx(G,
                    ax = axs[0], pos = pos,
                    with_labels = False, node_size = 28)
    axs[0].set_aspect('equal', adjustable='box')
    axs[0].axis('off')

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

    # 右子图
    sns.heatmap(A, cmap = 'Blues',
               ax = axs[1], annot = True, fmt = '.0f',
               xticklabels = list(G.nodes), yticklabels = list(G.nodes),
               linecolor = 'k', square = True, linewidths = 0.2, cbar = False)

    plt.savefig(fig_title + '.svg')

# 创建完全图
G_complete = nx.complete_graph(9)


# 可视化完全图
visualize(G_complete, '完全图')

# 创建图G，随机删除完全图中一半边
G = G_complete.copy(as_view=False)
# 副本，非视图
random.seed(8)
edges_removed = random.sample(list(G.edges), 18)
G.remove_edges_from(edges_removed)

visualize(G, '图G')

G_complement = nx.complement(G)
# 补图
visualize(G_complement, '图G补图')

```

代码 4. 补图、完全图三者的邻接矩阵 |  Bk6\_Ch18\_03.ipynb

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](mailto:jiang.visualize.ml@gmail.com)

## 有权重

对于有权重无向图，其邻接矩阵  $A$  的每个元素直接换成边的权重值即可。比如，图 7 这幅加权无向图对应的邻接矩阵为

$$A = \begin{bmatrix} 0 & 10 & 50 & 0 \\ 10 & 0 & 20 & 30 \\ 50 & 20 & 0 & 40 \\ 0 & 30 & 40 & 0 \end{bmatrix} \quad (8)$$

阅读完本章后文，大家会发现成对距离矩阵、成对亲近度矩阵、协方差矩阵等都可以看成是邻接矩阵；这也意味着这些矩阵都可以看成是图。

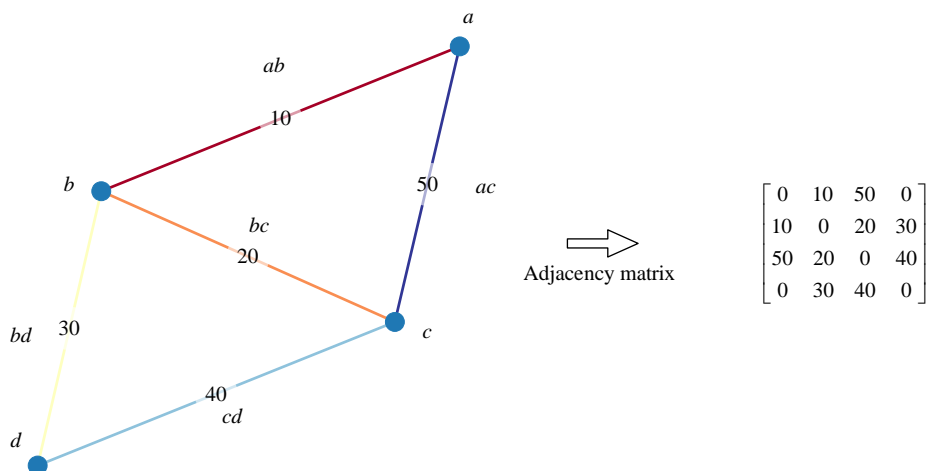


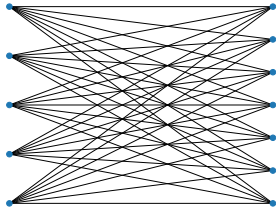
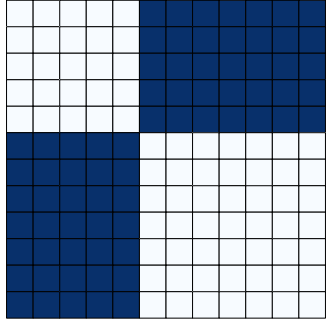
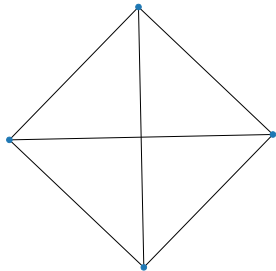
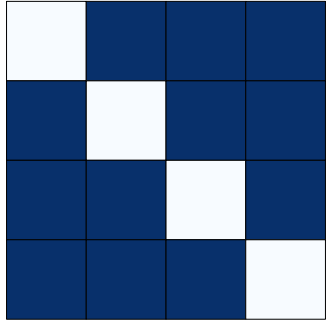
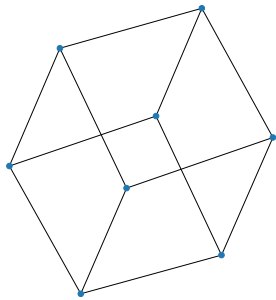
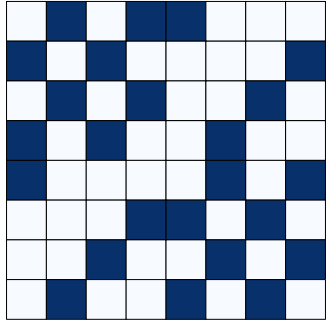
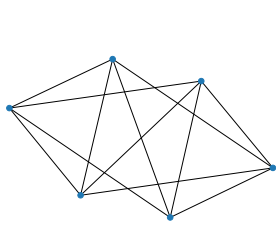
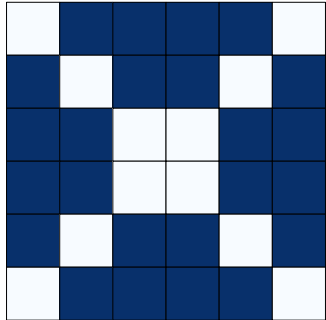
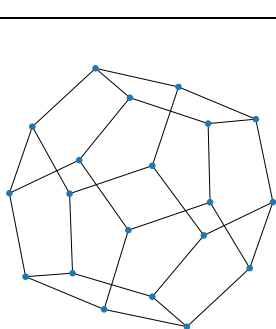
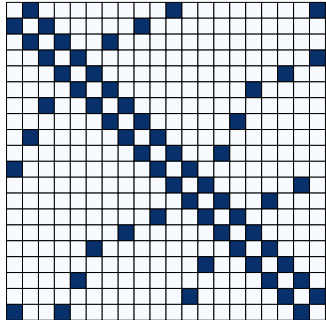
图 7. 加权无向图

## 特殊图的邻接矩阵

本书前文介绍了一些特殊的图，表 2 总结常见图及其邻接矩阵，请大家注意分析邻接矩阵展现出来的规律。Bk6\_Ch18\_04.ipynb 绘制表 2，请大家自行学习。

表 2. 常见图及其邻接矩阵

常见图	图	邻接矩阵
完全图		

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

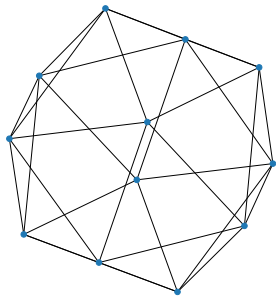
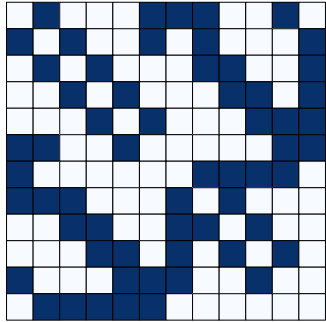
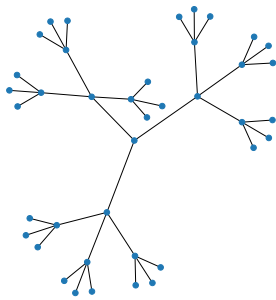
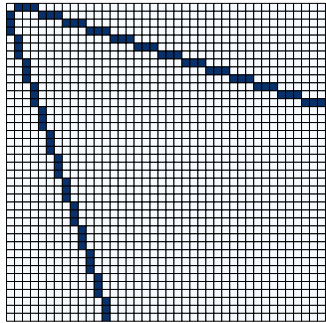
本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

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](mailto:jiang.visualize.ml@gmail.com)

正二十面体图		
平衡树		

## 18.2 有向图到邻接矩阵

理解了如何将无向图转换成邻接矩阵，就很容易掌握如何将 $\text{有向图}$ 转换成邻接矩阵。

不考虑权重的条件下，对于一个有向图  $G_D$ ，其邻接矩阵  $A$  的定义如下：

如果存在节点  $i$  到节点  $j$  之间有向边  $ij$ ，则  $a_{ij}$  的值为 1；

如果不存在节点  $i$  到节点  $j$  之间有向边  $ij$ ，则  $a_{ij}$  的值为 0。

请大家格外注意节点  $i$  到节点  $j$  的先后顺序。

有向图的邻接矩阵  $A$  一般不是对称矩阵。

图 8 所示的有向图对应的邻接矩阵为

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (9)$$

由于图 8 这幅有向图有 4 个节点，因此其邻接矩阵的形状也是  $4 \times 4$ 。和无向图一致，邻接矩阵  $A$  的 4 行从上到下分别代表 4 个节点—— $a$ 、 $b$ 、 $c$ 、 $d$ 。邻接矩阵  $A$  的 4 列从左到右分别代表这 4 个节点。

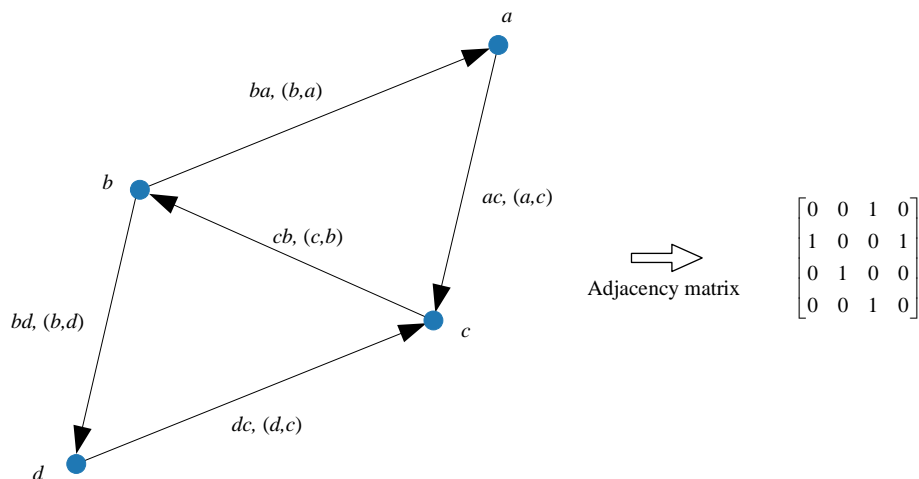


图 8. 从有向图到邻接矩阵，无自环

图 9 逐个元素解释有向图和邻接矩阵之间的关系。

举个例子，存在节点  $b$  到节点  $a$  的有向边  $ba$ ，因此邻接矩阵  $A$  中  $a_{2,1}$  元素为 1。反过来，由于不存在节点  $a$  到节点  $b$  的有向边  $ab$ ，因此邻接矩阵  $A$  中  $a_{1,2}$  元素为 0。请大家自行分析图 9 剩余几幅子图。

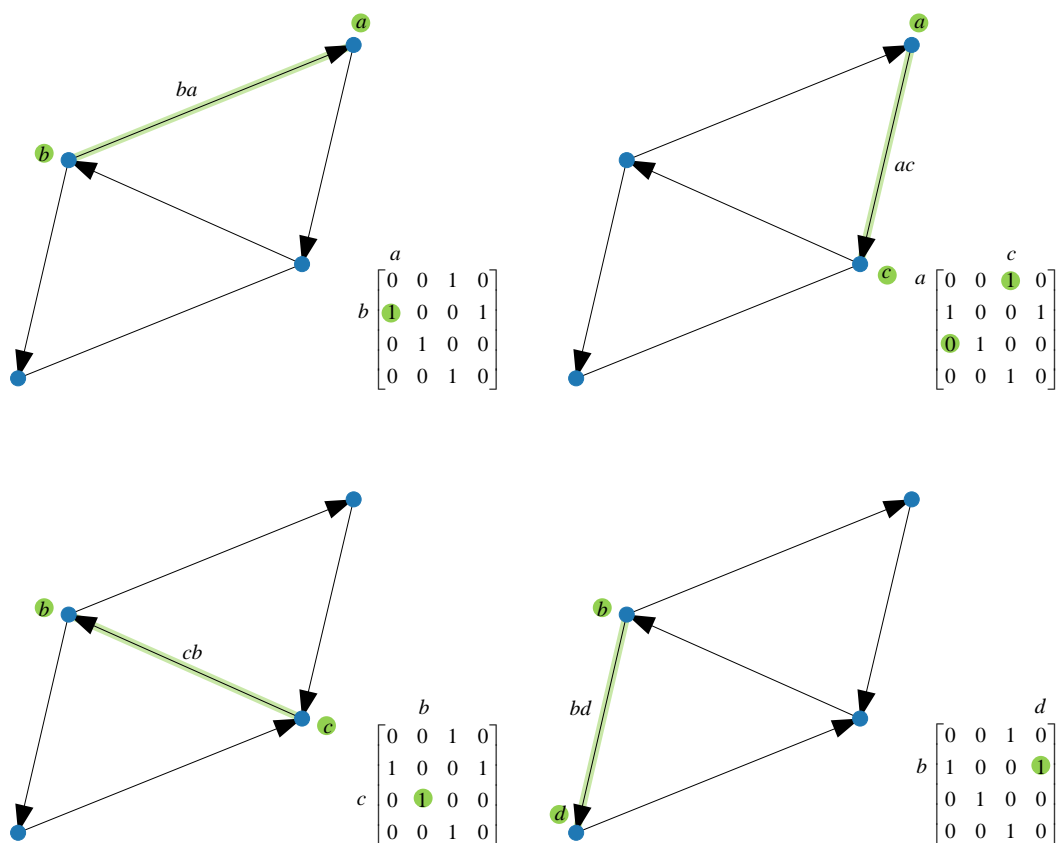


图 9. 逐个元素解释有向图和邻接矩阵之间的关系

### 有向图的邻接矩阵 $A$ 沿列方向求和为各个节点入度

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](mailto:jiang.visualize.ml@gmail.com)

$$\begin{bmatrix} 1 & 1 & 2 & 1 \end{bmatrix} \quad (10)$$

有向图的邻接矩阵  $A$  沿行方向求和为各个节点出度

$$\begin{bmatrix} 1 \\ 2 \\ 1 \\ 1 \end{bmatrix} \quad (11)$$

请大家自行学习 Bk6\_Ch18\_05.ipynb。

## 18.3 传球问题

邻接矩阵可以用来解决很多有趣的数学问题，本节举个例子。

有  $a$ 、 $b$ 、 $c$ 、 $d$ 、 $e$ 、 $f$  六名同学相互之间传球一只球。规则是，某个人每次传球可以传给其他任何人，但是不能传给自己。从  $a$  开始传球，传球 4 次，球最终回到  $a$  的手中，请大家计算一共有多少种传法。

图 10 所示为一种传法，传球路线为  $a \rightarrow f \rightarrow e \rightarrow b \rightarrow a$ 。

而图 11 展示的是回答传球问题的所有路径，下面的任务就是想办法完成计算。

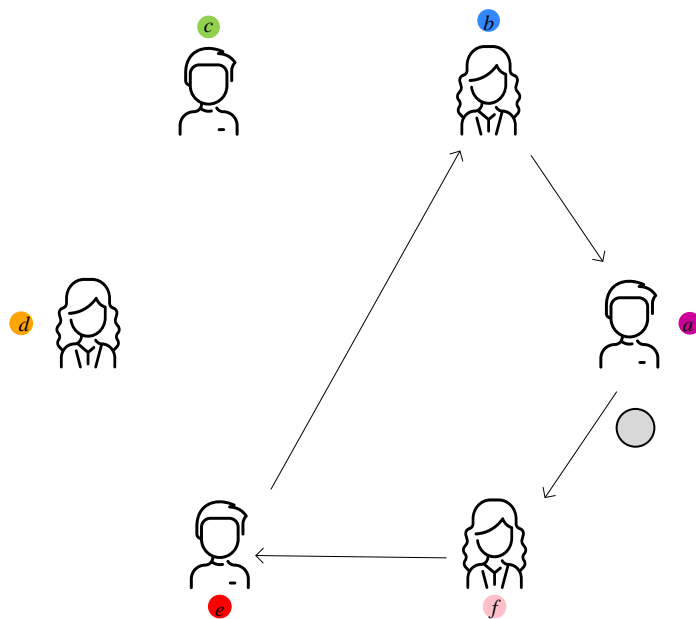


图 10. 一种传法



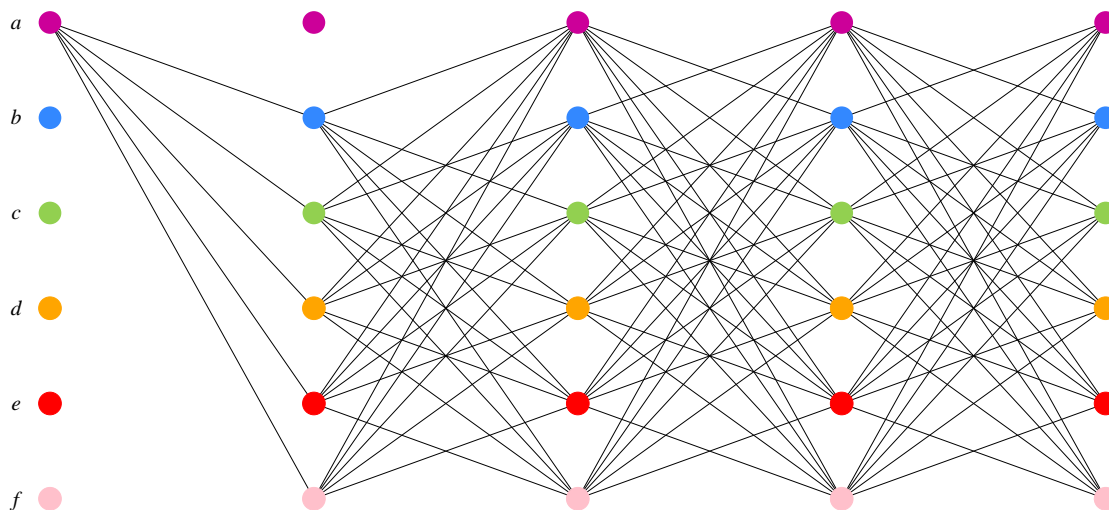


图 11. 所有可能路径的网络

把  $a$ 、 $b$ 、 $c$ 、 $d$ 、 $e$ 、 $f$  六名同学看成是六个节点的话，他们之间的传球关系可以抽象成图 12 所示有向图。而这幅有向图的邻接矩阵  $A$  为

$$A = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix} \quad (12)$$

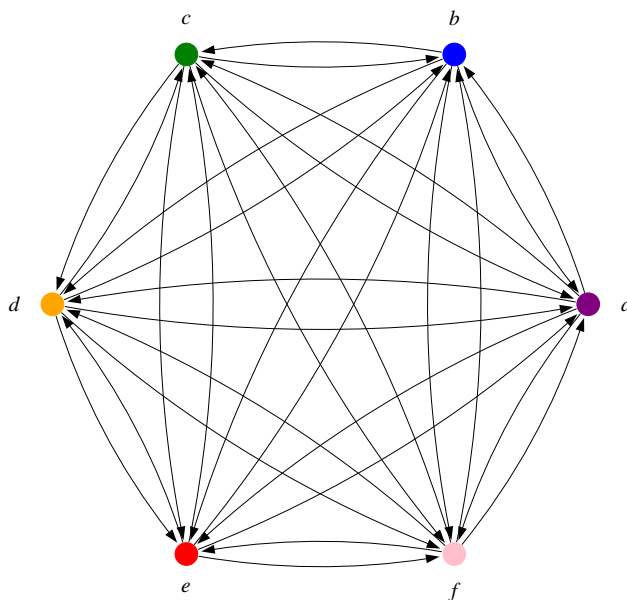


图 12. 代表传球问题的有向图

下面聊聊如何利用邻接矩阵  $A$  求解这个传球问题。

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](mailto:jiang.visualize.ml@gmail.com)

## 第 1 次传球

球最开始在  $a$  同学手里，将这个状态写成  $\mathbf{x}_0$

$$\mathbf{x}_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (13)$$

而矩阵乘法  $\mathbf{Ax}_0$  代表， $a$  同学手里的球在第 1 次传球后几种路径，具体结果为

$$\mathbf{x}_1 = \mathbf{Ax}_0 = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix} @ \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad (14)$$

如图 13 所示，这个结果表示，经过一次传球后，球可以在除了  $a$  之外的另外五名同学手上，也就是五种路径。这也是第 2 次传球的起点。

将向量  $\mathbf{x}_1$  的所有元素求和结果为 5。这个 5 实际上代表了  $5^1$ ，相当于一次传球后“一生五”。

看到 (14)，大家是否觉得“似曾相识”？我们在《数学要素》最后一章虚构“鸡兔互变”介绍马尔科夫链时也见过类似的矩阵乘法结构；而当时用到的方阵是**转移矩阵** (transition matrix)，如图 14 所示。本书后文会深入探讨邻接矩阵和转移矩阵之间的联系。

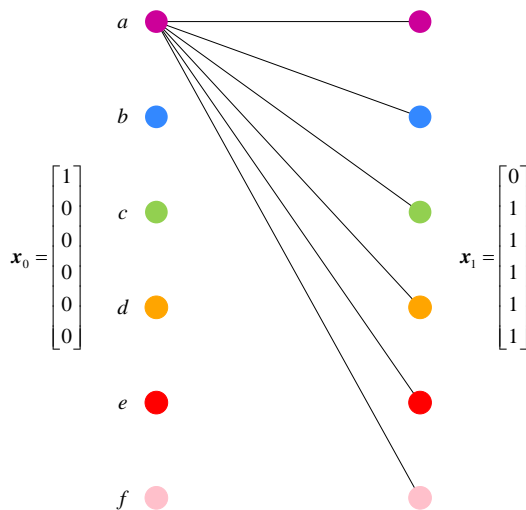


图 13. 矩阵乘法  $\mathbf{x}_1 = \mathbf{Ax}_0$  代表的具体含义

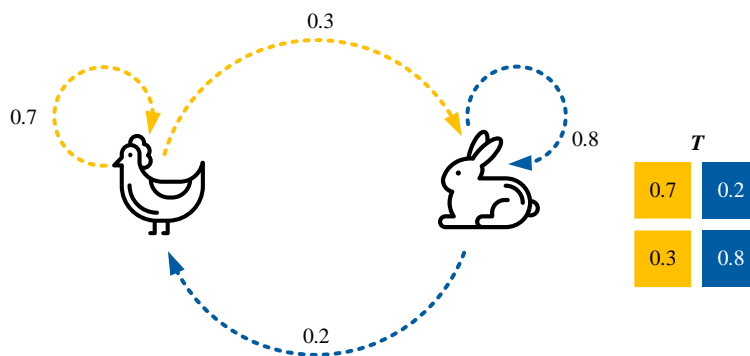
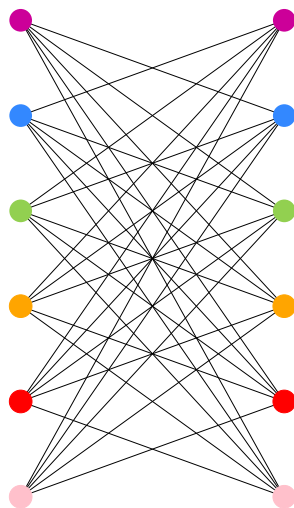


图 14. 鸡兔同笼三部曲中“鸡兔互变”，图片来自本系列丛书《数学要素》第 25 章

矩阵  $A$  所有元素求和的结果为 30，即  $6 \times 5$  (6 代表 6 个节点，5 代表每个节点有 5 条路径)。图 15 展示了这 30 条路径。

图 15. 矩阵  $A$  所有元素求和

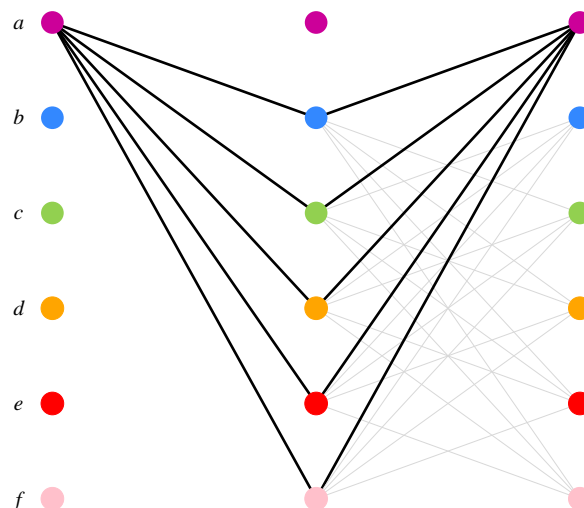
## 第 2 次传球

如图 16 所示，矩阵乘法  $A\mathbf{x}_1$  代表， $a$  同学手里的球在第 2 次传球后几种路径，具体结果为

$$\mathbf{x}_2 = A\mathbf{x}_1 = AA\mathbf{x}_0 = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix} @ \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 5 \\ 4 \\ 4 \\ 4 \\ 4 \\ 4 \end{bmatrix} \quad (15)$$

举个例子，向量  $\mathbf{x}_2$  的第 1 个元素为 5，这代表着 2 次传球后球回到  $a$  手上有 5 条路径。类似地，向量  $\mathbf{x}_2$  的第 2 个元素为 4，这代表着 2 次传球后球回到  $b$  手上有 4 条路径。

向量  $\mathbf{x}_2$  的所有元素求和结果为 25，代表了  $5^2$ ，相当于 2 次传球后“一生五、五生二十五”。

图 16. 矩阵乘法  $x_2 = Ax_1$  代表的具体含义

细心的读者可能已经发现，(15) 中核心运算是方阵  $A$  的幂，即  $A^2$ 。而  $A^2$  的结果具体为

$$A^2 = AA = \begin{bmatrix} 5 & 4 & 4 & 4 & 4 & 4 \\ 4 & 5 & 4 & 4 & 4 & 4 \\ 4 & 4 & 5 & 4 & 4 & 4 \\ 4 & 4 & 4 & 5 & 4 & 4 \\ 4 & 4 & 4 & 4 & 5 & 4 \\ 4 & 4 & 4 & 4 & 4 & 5 \end{bmatrix} \quad (16)$$

而 (15) 仅仅是取出  $A^2$  结果的第 1 列。

换个角度，如果修改本节题目，将初始持球者换成其他同学，我们仅仅需要修改初始状态向量  $x_0$

$$x_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad (17)$$

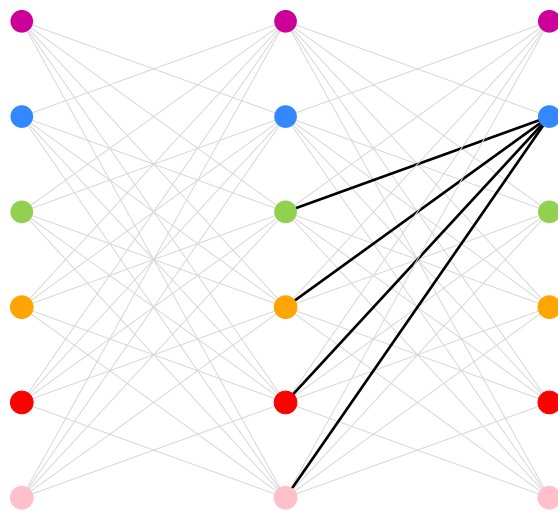
而对于不同初始状态向量  $x_0$ ， $A^2 x_0$  运算结果就是提取  $A^2$  的不同列。

$A^2$  结果也很值得细看！

$A^2$  的主对角线都是 5，这代表着经过两次传球，从某位同学手中再回到本人的路径。

除了主对角线元素之外， $A^2$  其他元素都是 4。出现这个结果也不意外。举个例子，开始时如果球在  $a$  手中，两次传球后球在  $b$  手中有 4 种路径。由于  $b$  不能传给自己，这刨除一条路径。此外， $a$  不能传给自己，然后再传给  $b$ ，这又刨除了一条路径。实际上，这是利用组合数求解这个问题的内核。

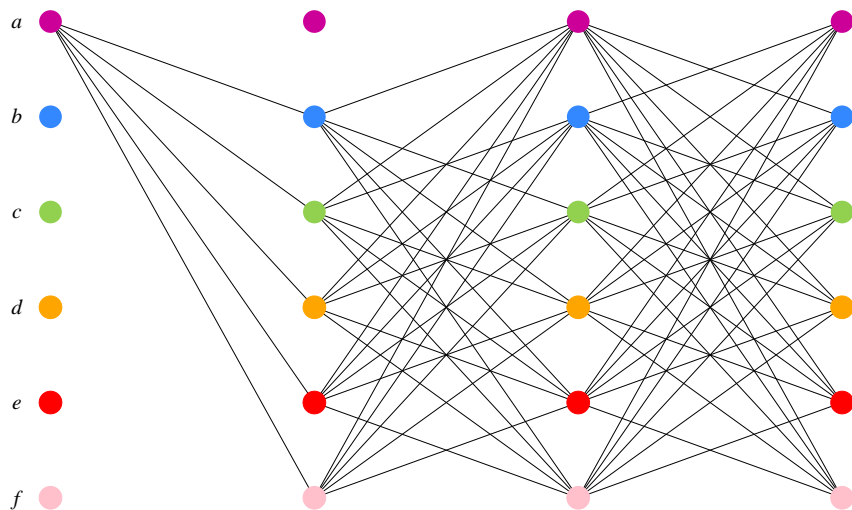
而  $A^2$  的所有元素之和为 150，即  $6 \times 5 \times 5$ 。

图 17. 方阵乘幂  $A^2$  代表的具体含义

### 第 3 次传球

如图 18 所示，矩阵乘法  $Ax_2$  代表， $a$  同学手里的球在第 3 次传球后几种路径，具体结果为

$$x_3 = Ax_2 = AAAx_0 = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix} @ \begin{bmatrix} 5 \\ 4 \\ 4 \\ 4 \\ 4 \\ 4 \end{bmatrix} = \begin{bmatrix} 20 \\ 21 \\ 21 \\ 21 \\ 21 \\ 21 \end{bmatrix} \quad (18)$$

图 18. 矩阵乘法  $x_3 = Ax_2$  代表的具体含义

而  $A^3$  的结果具体为

$$A^3 = AAA = \begin{bmatrix} 20 & 21 & 21 & 21 & 21 & 21 \\ 21 & 20 & 21 & 21 & 21 & 21 \\ 21 & 21 & 20 & 21 & 21 & 21 \\ 21 & 21 & 21 & 20 & 21 & 21 \\ 21 & 21 & 21 & 21 & 20 & 21 \\ 21 & 21 & 21 & 21 & 21 & 20 \end{bmatrix} \quad (19)$$

(18) 相当于取出 (19) 的第 1 列。

请大家自行分析为什么  $A^3$  的主对角线元素为 20，而其他元素为 21。

#### 第 4 次传球

矩阵乘法  $Ax_3$  代表， $a$  同学手里的球在第 4 次传球后几种路径，具体结果为

$$x_4 = Ax_3 = A^4 x_0 = \begin{bmatrix} 105 \\ 104 \\ 104 \\ 104 \\ 104 \\ 104 \end{bmatrix} \quad (20)$$

上式告诉我们本节最开始提出的问题答案为 105。

而  $A^4$  的结果具体为

$$A^4 = \begin{bmatrix} 105 & 104 & 104 & 104 & 104 & 104 \\ 104 & 105 & 104 & 104 & 104 & 104 \\ 104 & 104 & 105 & 104 & 104 & 104 \\ 104 & 104 & 104 & 105 & 104 & 104 \\ 104 & 104 & 104 & 104 & 105 & 104 \\ 104 & 104 & 104 & 104 & 104 & 105 \end{bmatrix} \quad (21)$$

请大家思考，如果传球四次，从  $a$  开始传球，球最终回到  $f$  手中，共有多少种传法。此外，请大家自行思考如何用组合数求解这个问题。最后，如果修改传球规则，允许将球传给自己，这对有向图和邻接矩阵有何影响。

Bk6\_Ch18\_04.ipynb 完成本节传球问题的具体编程实践，下面聊聊其中关键语句。

- a** 用 `networkx.complete_nodes()` 生成有向完全图。
- b** 定义字典，用来替换默认节点名称。
- c** 定义列表，其中包含节点颜色名称。
- d** 用 `networkx.relabel_nodes()` 重新定义节点名称，输入中用到了前面定义的映射字典。
- e** 用 `networkx.circular_layout()` 创建圆周布局位置坐标。
- f** 绘制有向图，其中用参数 `connectionstyle` 规定了用弧线方式展示有向边。
- g** 生成有向图的邻接矩阵。
- h** 定义初始向量，代表球在  $a$  同学手中。

Bk6\_Ch18\_04.ipynb 还给出用组合数求解传球问题的方法。

```

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

a G = nx.complete_graph(6, nx.DiGraph())

b mapping = {0: 'a', 1: 'b', 2: 'c', 3: 'd', 4: 'e', 5: 'f'}
c node_color = ['purple', 'blue', 'green', 'orange', 'red', 'pink']
d G = nx.relabel_nodes(G, mapping)
e pos = nx.circular_layout(G)

# 可视化
plt.figure(figsize = (6,6))
f nx.draw_networkx(G,
                    pos = pos,
                    connectionstyle='arc3, rad = 0.1',
                    node_color = node_color,

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

# 球在A手里
h x0 = np.array([[1,0,0,0,0,0]]).T

# 第1次传球
x1 = A @ x0

# 第2次传球
x2 = A @ x1

# 第3次传球
x3 = A @ x2

# 第4次传球
x4 = A @ x3

# 矩阵A的4次幂
A@A@A@A

```

代码 5. 传球问题 |  Bk6\_Ch18\_06.ipynb

## 18.4 邻接矩阵的矩阵乘法

有向图的邻接矩阵乘法蕴含很多关于图的信息，本节简单总结一下。

本节用到的有向图如图 19 所示。这幅有向图中，节点  $a$ 、 $b$  和节点  $b$ 、 $c$  之间各有一对方向相反的有向边。

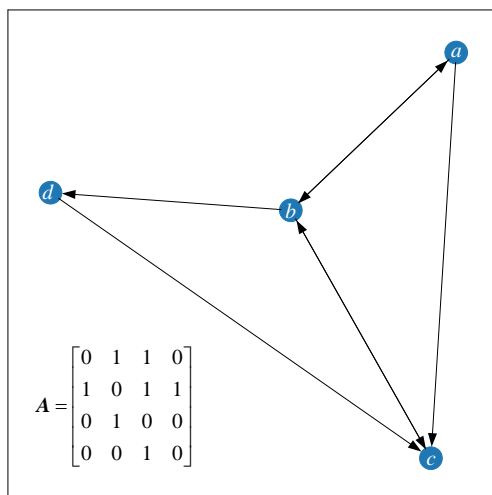


图 19. 展示邻接矩阵乘法的有向图

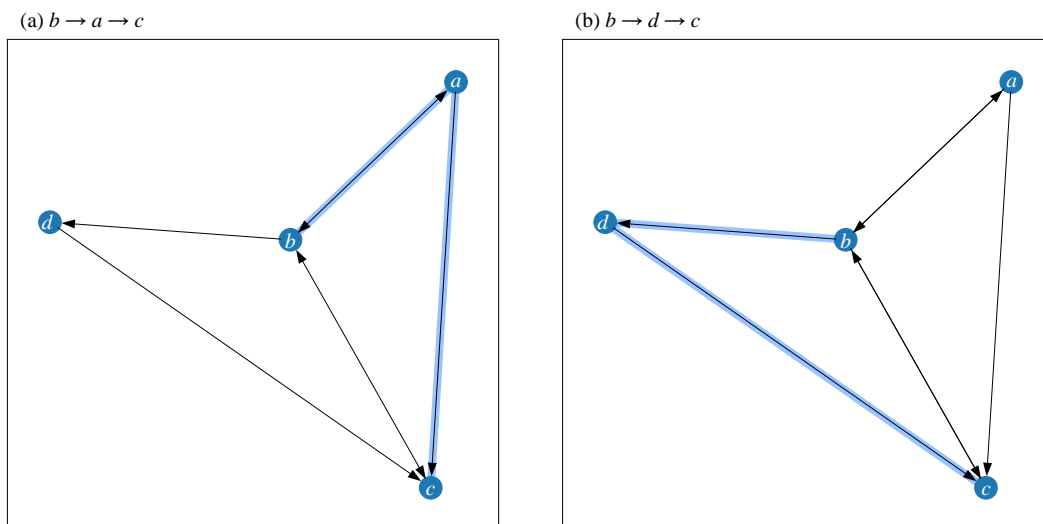
 $A^2$ 

通过上节传球的例子，我们已经知道邻接矩阵的平方  $A^2$  可以帮我们节点之间长度为 2 的路径。图 19 中有向图的邻接矩阵的平方为

$$A^2 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 2 & 2 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix} \quad (22)$$

比如上述矩阵第 2 行、第 3 列元素代表从  $b$  到  $c$  长度为 2 的路径数量有 2 条，如图 20 所示。

同理， $A^3$  可以帮我们节点之间长度为 3 的路径； $A^n$  ( $n$  为正整数) 表示节点之间长度为  $n$  的路径。

图 20. 从  $b$  到  $a$  长度为 2 的路径数量有 2 条 $A @ A^T$ 

邻接矩阵  $A$  乘自身转置  $A^T$  的结果为

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](mailto:jiang.visualize.ml@gmail.com)



$$A @ A^T = \begin{bmatrix} 2 & 1 & 1 & 1 \\ 1 & 3 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix} \quad (23)$$

显然， $A @ A^T$  为对称矩阵，因为  $A @ A^T$  是格拉姆矩阵。

$A @ A^T$  的对角线元素为节点的出度；比如，节点  $a$  的出度为 2，节点  $b$  的出度为 3，具体如图 21 所示。

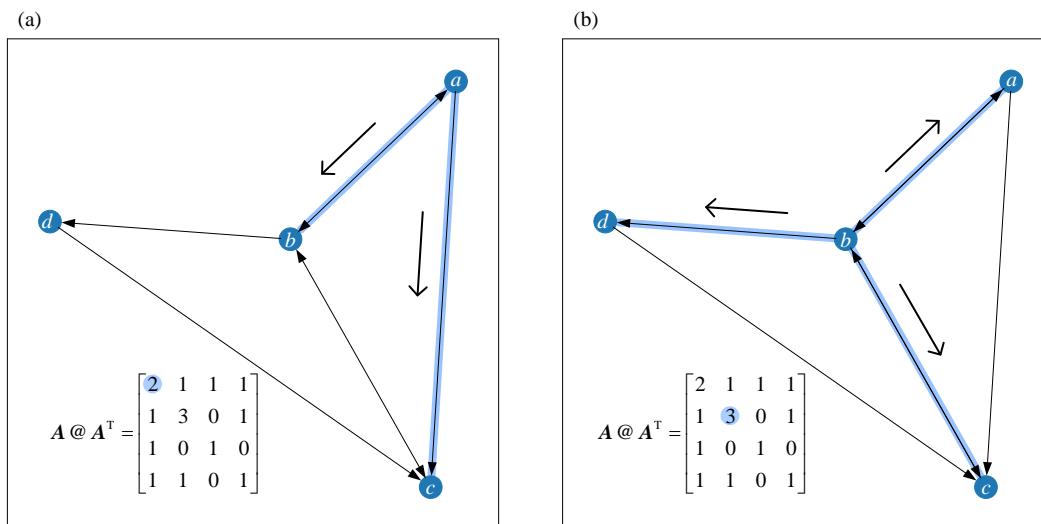


图 21. 有向图节点出度

$A @ A^T$  的非对角线元素代表某对节点引出指向同一节点的节点数。

比如，节点  $a$ 、 $b$  都有一条指向节点  $c$  的有向边，具体如图 22 (a) 所示。

图 22 (b) 则展示节点  $b$ 、 $d$  都有一条指向节点  $c$  的有向边。

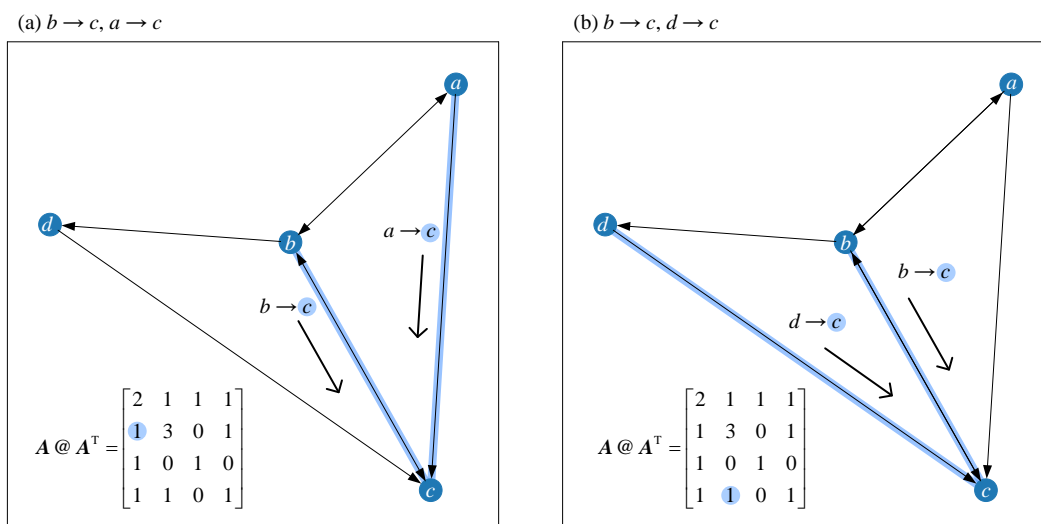


图 22. 节点  $a$ 、 $b$  都有一条指向节点  $c$  的有向边，节点  $b$ 、 $d$  都有一条指向节点  $c$  的有向边

$A^T @ A$ 

邻接矩阵转置  $A^T$  乘自身  $A$  的结果为

$$A^T @ A = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 2 & 1 & 0 \\ 1 & 1 & 3 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix} \quad (24)$$

$A^T @ A$  也是格拉姆矩阵；因此， $A^T @ A$  是对称矩阵。

$A^T @ A$  的对角线元素为节点的入度；比如，节点  $a$  的入度为 1，节点  $b$  的入度为 2，具体如图 23 所示。

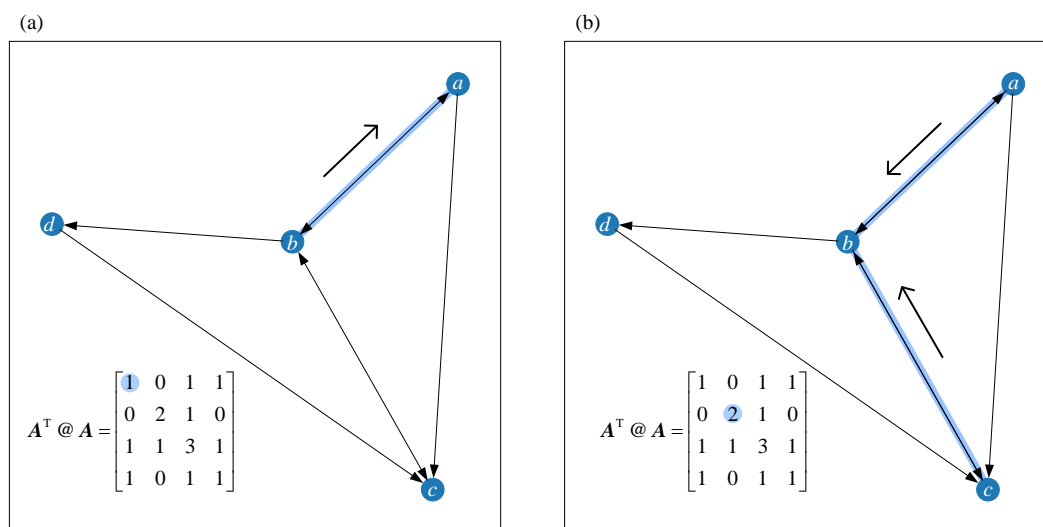


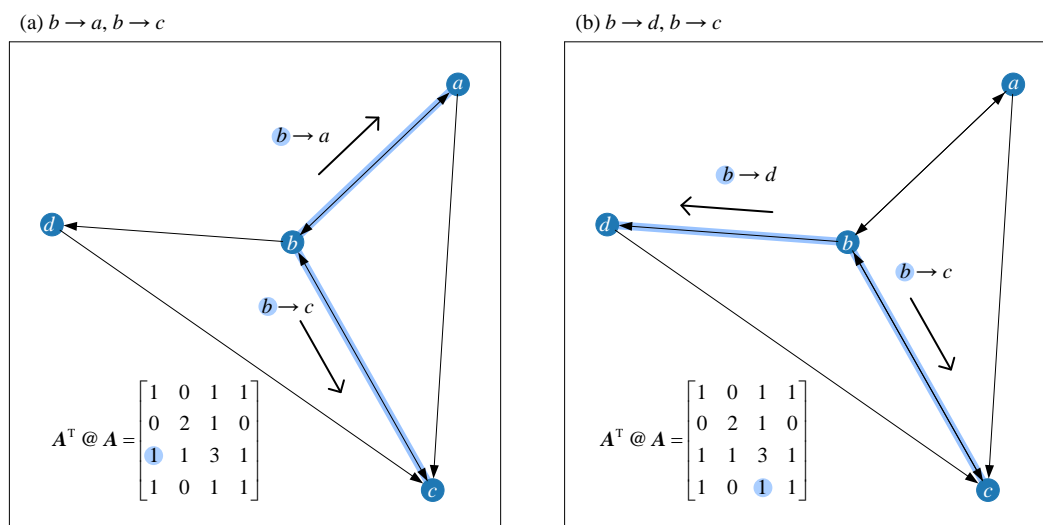
图 23. 有向图节点入度

和  $A @ A^T$  相反， $A^T @ A$  的非对角线元素代表同时指向特定节点的节点数。

比如，节点  $b$  有两条分别指向节点  $a$ 、 $c$  的有向边，具体如图 24 (a) 所示。

图 24 (b) 则展示节点  $b$  有两条分别指向节点  $c$ 、 $d$  的有向边。

本节相关运算都在 Bk6\_Ch18\_07.ipynb，请大家自行学习。



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](mailto:jiang.visualize.ml@gmail.com)

图 24. 节点  $b$  有两条分别指向节点  $a$ 、 $c$  的有向边，节点  $b$  有两条分别指向节点  $c$ 、 $d$  的有向边

## 18.5 特征向量中心性

本书前文介绍了几种**中心性度量** (centrality measure):

- ▶ **度中心性** (degree centrality) 用节点的度数描述其“中心性”。
- ▶ **介数中心性** (betweenness centrality) 用于衡量一个节点在图中承担“桥梁”角色的程度。
- ▶ **紧密中心性** (closeness centrality) 节点平均最短距离的倒数。

本节再介绍一种基于邻接矩阵的中心度——**特征向量中心性** (eigenvector centrality)。

Bk6\_Ch18\_08.ipynb 用空手道俱乐部人员关系图为例，用 `networkx.eigenvector_centrality()` 计算每个节点的特征向量中心性度量值；并且根据度量值大小渲染无向图节点。如图 25 (b) 所示，暖色代表特征向量中心性度量值大，冷色则相反。显然，节点 0、33 的中心性度量值最高。这个本书前文其他中心性度量值结论一致。

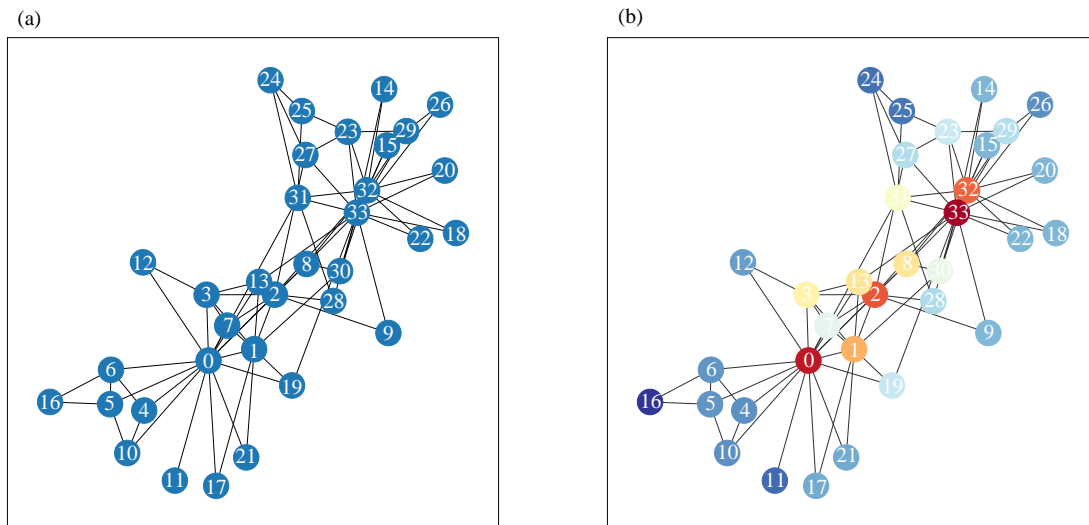


图 25. 空手道俱乐部人员关系图，根据特征向量中心性度量值渲染节点

Bk6\_Ch18\_08.ipynb 还给出图 26 这个更复杂的图例，节点颜色、大小都是根据特征向量中心性值大小决定的。图 27 所示为特征向量中心性度量值的分布。

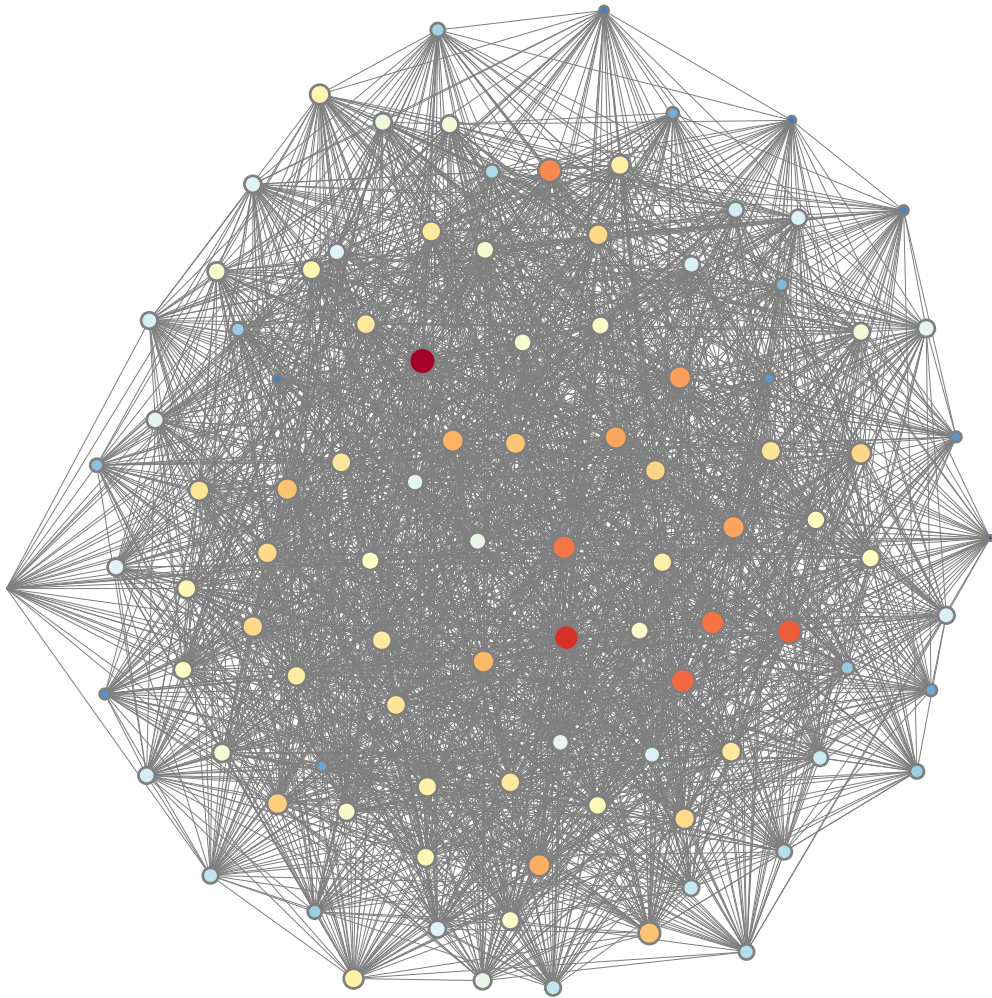


图 26. 根据特征向量中心性大小渲染节点

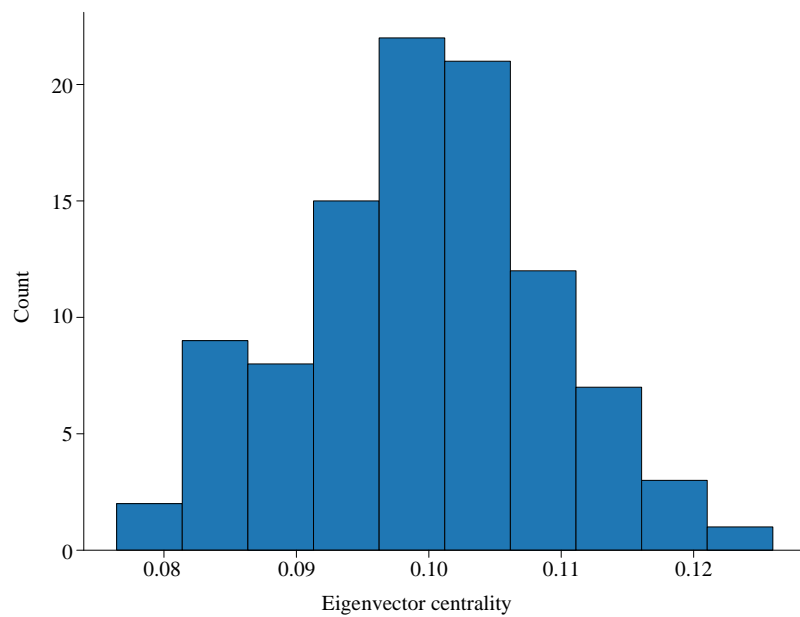
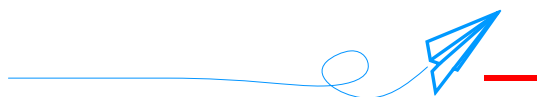


图 27. 特征向量中心性的分布



图论中的邻接矩阵是一种表示图中各节点之间相互连接关系的矩阵。对于一个有  $n$  个节点的图，邻接矩阵是一个  $n \times n$  的矩阵，其中的元素定义了节点间是否存在边。对于无向图，邻接矩阵是对称的。简单图的邻接矩阵矩阵中的元素通常是 0 或 1，其中 1 表示两个节点之间存在边，而 0 表示不存在边。在加权图中，邻接矩阵元素代表边的权重。

下一章，大家会发现成对距离矩阵、亲近度矩阵、协方差矩阵、相关性系数矩阵等等都可以看做是邻接矩阵；也就是说，这些矩阵都可以看做是图！

鸢尾花书的读者应该还记贯穿《矩阵力量》始终的这几句话。

有数据的地方，必有矩阵！

有矩阵的地方，更有向量！

有向量的地方，就有几何！

有几何的地方，皆有空间！

有数据的地方，定有统计！

学完本书后，我们还要再加上一句：

图就是矩阵，矩阵就是图。