1 Making and Recognizing Networks

1.1 What Is a Network?

Networks are made up of **nodes** and **edges**. Nodes (or **vertices**) often represent people, places, or things. Edges represent relationships or **links** between nodes. If we have a list of nodes (N) and edges (E), then we have a network. Networks are sometimes referred to as **graphs** and indicated by $G(N, E)$.

The network in Figure 1.1 has six nodes and eight edges.

We can imagine there are six people and eight relationships. Node 6 has four friends and one of them is Node 2. But there is no reason why these have to be people and their relationships. Nodes could represent words and edges could represent shared meanings. They could be places and paths between them, symptoms and comorbidity, brands and shared product lines, organisms and shared environments, beliefs and their co-occurrence in the general population. Whenever we have 'things' (in the broadest sense of the term) and relationships, we have nodes and edges. Then we have a network.

1.2 Representing Networks

We can represent the node and edge relationships in two distinct ways: either as an edge list or an adjacency matrix. An **edge list** lists each relationship: there is a row for every edge, listing the names of the vertices in two columns. The list in Table 1.1 corresponds to the eight edges shown in Figure 1.1. V1 and V2 correspond to the two vertices that are connected by an edge.

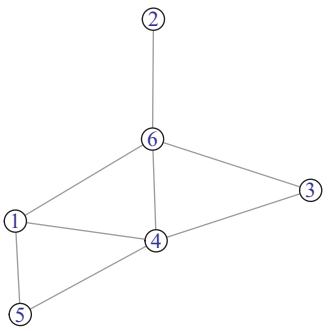

Figure 1.1 A basic network.

Table 1.1 An edgelist for our basic network.

V1	V2
1	4
3	4
1	5
4	5
1	6
2	6
3	6
4	6

Table 1.2 An adjacency matrix for our basic network.

	1	2	3	4	5	6
1	0	0	0	1	1	1
2	0	0	0	0	0	1
3	0	0	0	1	0	1
4	1	0	1	0	1	1
5	1	0	0	1	0	0
6	1	1	1	1	0	0

The first edge in the edge list is between Node 1 and Node 4. This corresponds to an edge we can see in the visualization. The edge list is not in any particular order from top to bottom.

We can also represent a network using an **adjacency matrix**, which is a matrix indicating which nodes are connected by edges. Table 1.2 provides an example.

For every edge in the edge list, there is a nonzero value in the corresponding cell of the adjacency matrix. The value of the edge between node i and node j is represented by A_{ij}. Row 4, column 1 has a value of 1: $A_{4,1} = 1$, indicating the presence of an edge. Note that $A_{1,4}$ also has a 1. Whenever we have an **undirected network** – meaning that edges do not indicate any form of directionality – then the adjacency matrix is symmetric: $A_{ij} = A_{ji}$.

The **diagonal** of the matrix refers to all cells A_{ij} for which $i = j$. In other words, they represent **self-loops**. If no nodes connect to themselves, the diagonal is zero. We can add a self-loop for Node 2 by adding an edge from 2 to itself: $A_{2,2} = 1$, as shown in Table 1.3 and Figure 1.2.

Self-loops are infrequent in behavioral network science. But they are sometimes useful. For example, if edges represent people who share cake with one another, then we might imagine that some people eat some of their own cake. Those people would have a self-loop.

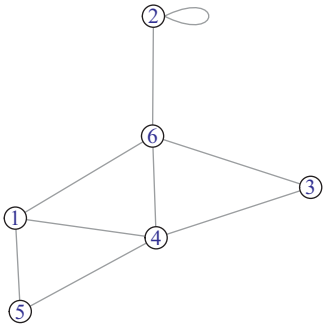

Figure 1.2 A network with a self-loop.

Table 1.3 Adjacency matrix for a network with a self-loop.

	1	2	3	4	5	6
1	0	0	0	1	1	1
2	0	1	0	0	0	1
3	0	0	0	1	0	1
4	1	0	1	0	1	1
5	1	0	0	1	0	0
6	1	1	1	1	0	0

1.3 Four Types of Networks

The network described in Figure 1.1 is the simplest of networks: Edges have values of 0 or 1 and are undirected. There is only one kind of node and there is only one kind of edge. It's all very basic. All other types of networks diverge from such basic networks by varying different properties of edges and nodes.[1]

1.3.1 Weighted Networks

When edges can take values different from 0 or 1, the network is **weighted**. Figure 1.3 shows a weighted network.

We might imagine that the eight relations are each valued according to the intensity of their love. Node 4 gets the most love. But the weights could just as easily be duration, frequency of contact, relative similarity, or number of shared features. Any relation that varies can be represented by a weighted edge.[2]

The corresponding weighted adjacency matrix contains edge weights in each cell, as shown in Table 1.4.

[1] The term **simple network** is reserved for networks with no self-loops and no multi-edges. Thus, pretty much all the networks we discuss here are simple by this definition.

[2] A slight variation is a **signed network**, which can represent positive (love) and negative (hate) relationships, useful for tracking bullying or international alliances.

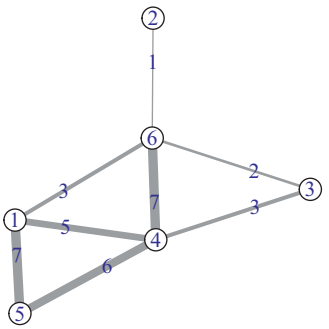

Figure 1.3 A weighted network. Edge thickness is proportional to edge weight.

Table 1.4 Weighted adjacency matrix.

	1	2	3	4	5	6
1	0	0	0	5	7	3
2	0	0	0	0	0	1
3	0	0	0	3	0	2
4	5	0	3	0	6	7
5	7	0	0	6	0	0
6	3	1	2	7	0	0

Table 1.5 Weighted edge list.

V1	V2	weight
1	4	5
3	4	3
1	5	7
4	5	6
1	6	3
2	6	1
3	6	2
4	6	7

We can write an edge list for the same network by adding the weights in a third column, as in Table 1.5.

1.3.2 Directed Networks

When edges point from one node to another, the network is a **directed network**, as shown in Figure 1.4. Now we can imagine our six nodes are sumo wrestlers. Directed edges represent victories. These could go either way, but here we will let the edge point from the loser to the victor. Nodes 6 and 5 are undefeated.

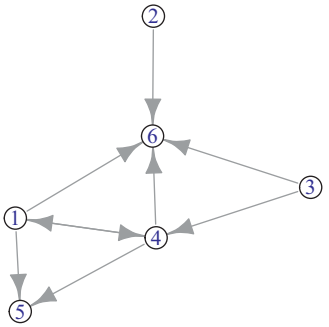

Figure 1.4 A directed network.

Table 1.6 Directed edge list.

V1	V2
1	4
3	4
1	5
4	5
1	6
2	6
3	6
4	6
4	1

Table 1.7 Directed adjacency matrix.

	1	2	3	4	5	6
1	0	0	0	1	1	1
2	0	0	0	0	0	1
3	0	0	0	1	0	1
4	1	0	0	0	1	1
5	0	0	0	0	0	0
6	0	0	0	0	0	0

For a directed network the edge list indicates which node points to which other node: The convention is the first column points to the second column. This creates a corresponding asymmetry in the adjacency matrix. The row node points to the column node. Compare the edge list in Table 1.6 with the adjacency matrix in Table 1.7.

When an edge points in both directions, $A_{ij} = A_{ji}$, we say it is a **reciprocal edge**. Nodes 4 and 1 in our sumo network share reciprocal wins.

Crossing unweighted with weighted and undirected with directed produces four network types. Most of the networks we deal with will be one of the following four

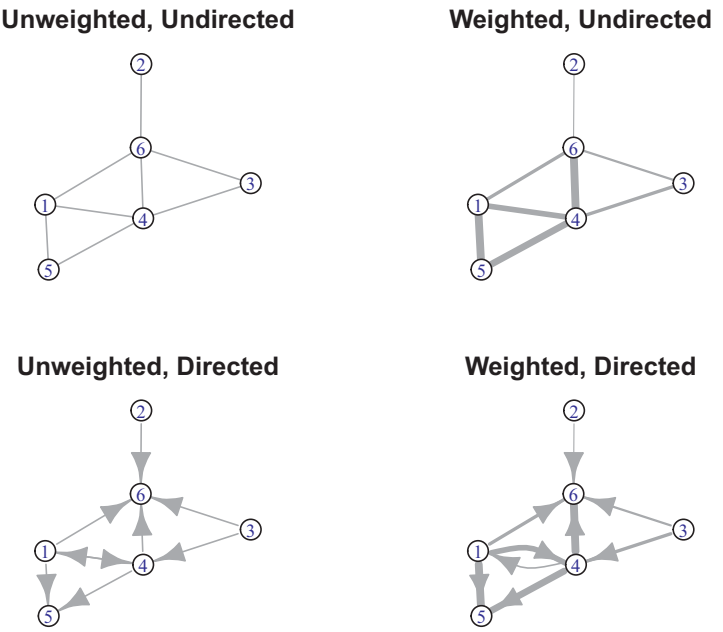

Figure 1.5 Four common network types.

(Figure 1.5): (1) unweighted and undirected, (2) weighted and undirected, (3) unweighted and directed, or (4) weighted and directed.

1.3.3 Sparse Matrices, Edge Lists, and Adjacency Matrices

Data often comes in the form of edge lists or adjacency matrices. For example, if we have people list their friends, we have an edge list. When the output of a similarity measure produces a large matrix of pairwise similarities, we have an adjacency matrix. One can quickly be turned into the other.

A savvy network scientist can quickly infer properties of a network by looking at the adjacency matrix. For example, is the network symmetric? Is it weighted? It is often useful to dummy check the adjacency matrix to make sure it reflects your understanding and to verify that edges are where and how you think they should be.

However, for networks with many nodes, the adjacency matrix can be unreasonably large. A directed network with 1,000 nodes might have 1,000,000 possible edges, with the same number of values in the adjacency matrix. As this grows, so grows the computational resources needed to handle it.

Luckily, most networks are **sparse matrices**, meaning that most of the edge values are 0. The number of nonzero edges is much smaller than the number of possible edges. This means that the edge list will be a much smaller computer memory object than the adjacency matrix. If our network with 1,000,000 possible edges had only one actual edge, we could represent it with one row in an edge list. Most network packages have methods for interacting with large but sparse matrices in fast and accurate ways.

1.4 Thresholding

Our basic network is often the easiest to study, visualize, and interpret. When it captures the relevant information necessary to address a particular problem, it is preferred to a weighted or directed network. When that is the case, it is useful to be able to transform a more complex network into a basic network. There are two traditional routes to achieve this.

1.4.1 Thresholding Weighted Networks

Suppose we have a network representing how often ministers in the UK Parliament vote together. We would have a weighted network with each edge representing the number of shared votes. If we plotted this as a network, every node would be connected to every other node because ministers occasionally vote unanimously. How might we turn this into a useful unweighted and undirected network? One approach is **thresholding**: only keep edges with weights above some threshold value. Here the ministers must share more than a threshold number of votes. Below this threshold we assign them an edge value of 0 and above it we assign a value of 1.

In Figure 1.6 there are two types of nodes (dark and light). Edges between nodes of the same color have larger weights. You cannot see this in the visualization if the threshold is 0; everything is connected. As the threshold increases, the two groups become visible. At too high a threshold, all nodes become isolates.

What is the optimal threshold? That depends on the problem. Best practice usually involves choosing multiple thresholds over a range that captures this transition and presenting the results for all of them. This reduces the worry that results are sensitive to an arbitrary threshold.

1.4.2 Thresholding Directed Networks

Suppose we ask a group of students in a classroom to tell us who their friends are. The resulting friendship network is directed, with arrows pointing toward hopefully reciprocated friendships. How might we turn this into an unweighted and undirected network?

An undirected edge can be defined for the two variations of directed edges: (a) Either at least one directed edge is present, or (b) the edges are reciprocal – each node directs

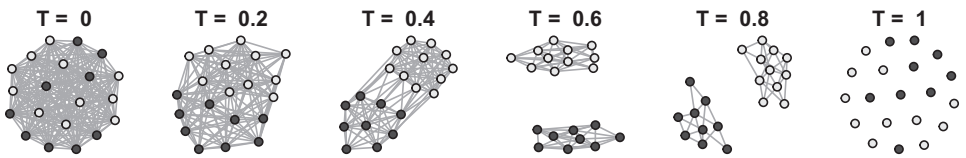

Figure 1.6 Thresholding weighted networks. Dark and light nodes have stronger weights with other members of their group. This is easiest to see when the threshold for edge weights is an intermediate value.

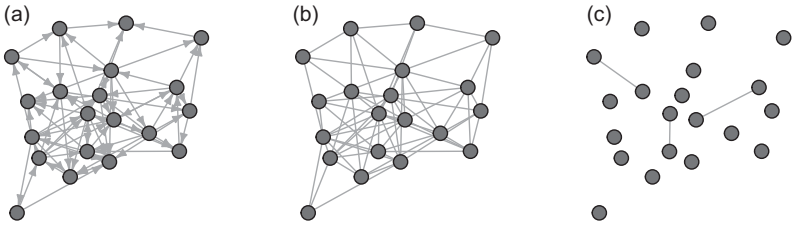

Figure 1.7 Thresholding directed networks. The network in panel (a) shows the full directed network. The panel (b) network transforms this into an undirected network, making an edge wherever at least one node has a directed edge to the other. The network in panel (c) only preserves reciprocal edges.

an edge to the other. Figure 1.7 shows a directed network and the products of applying the at-least-one-edge and reciprocal-edge rules.

Which approach is best will depend on the question. If the researcher is studying airborne disease spread, then the direction of the edge may not matter as long as the edge represents contact. But if the researcher is studying a potential outbreak of tuberculosis – which requires prolonged contact – then reciprocal edges may be more meaningful.

1.5 Attributes of Nodes and Edges

Network representations can be enriched further by assigning different properties to the nodes and edges. The **attributes** can be categorical (e.g., democrat or republican) or continuous (e.g., height in millimeters). If we are trying to make predictions about nodes or edges, we can think of these attributes as additional predictors in our model.

If nodes are people, they can have different political affiliations, social strategies, ages, socioeconomic status, ethnicity, and so on. If nodes are words, they could have different valences (positive or negative meaning), grammatical types (e.g., nouns or verbs), frequencies of use, levels of concreteness, and many other features. Edges can vary along different dimensions as well.

These attributes often form the basis of hypotheses. For example, do individuals of different political affiliations (node property) have different numbers of social contacts? Does the age of a friendship (edge property) influence the well-being of the individuals (nodes) who share that friendship? Figure 1.8 visualizes these relationships.

1.5.1 Node Attributes: Bipartite Networks

Bipartite networks occur when nodes of one type are only associated with nodes of another type. Sometimes one of the node types represents possible features or attributes that can be shared among nodes of the other type.

Bipartite networks have many uses. For example, if people choose behaviors (e.g., movies to watch), a bipartite network can be used as a recommender system to identify which movie to recommend next. Bipartite networks can also be used to determine which

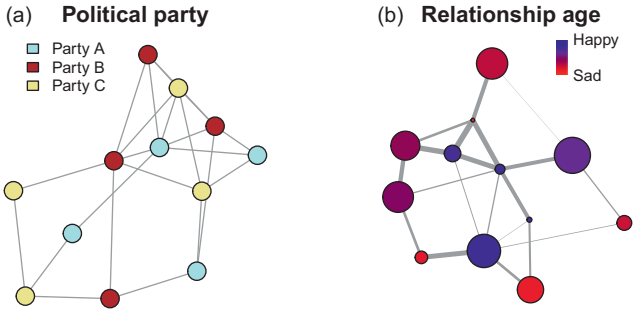

Figure 1.8 Categorical and continuous node attributes. (a) A network showing social contacts among representatives of different political parties. (b) A network of friendships. The edge width in the "Relationship age" network indicates the age of the relationship (an edge weight but also an edge attribute). Node size indicates age of the individual. Color indicates happiness.

individuals or behaviors are similar to one another, by looking at how nodes of one type share nodes of the other type.

Groups of nodes connected to a node in a bipartite network are functionally connected by their common affiliation. These are sometimes called **affiliation networks** or **hypergraphs**. In that sense, bipartite networks can be thought of as networks where edges connect more than two individuals at a time – for example, when individuals are members of clubs.

The two networks on the top of Figure 1.9 are both the same network. The one on the left is in a typical force-directed graph visualization.[3] The representation on the right is a bipartite projection, clearly showing the two node types. The bipartite projection shows how the nodes do not connect with nodes of the same type but only with nodes of the other type. For example, executives may be on multiple company boards, senators can vote for the same bills, people can share similar hobbies, and objects can share similar features.

The adjacency matrix of a bipartite network – often called an **incidence matrix** – assigns different types of nodes to the rows and columns of the matrix (see Table 1.8). Unlike the square monopartite matrix, with only one node type, the bipartite network can be represented as a rectangular adjacency matrix.

Bipartite networks can be **projected** onto one or the other node type. Nodes are then connected with one another if they share a connection with a node of the other type. The bottom two networks in Figure 1.9 are the one-mode (monopartite) projections for the two-mode (bipartite) network shown in the same figure.

1.5.2 Edge Attributes: Multiplex Networks

Edges in the same network can be of different types. Such **multiplex** (or **multilayer**) networks are commonly used to characterize systems in which nodes share different kinds of

[3] **Force-directed graph layouts** are physical simulations of the network that often treat edges as springs and nodes as repulsive electrostatically charged particles. The forces can be tuned to enhance the aesthetic features of the visualization.

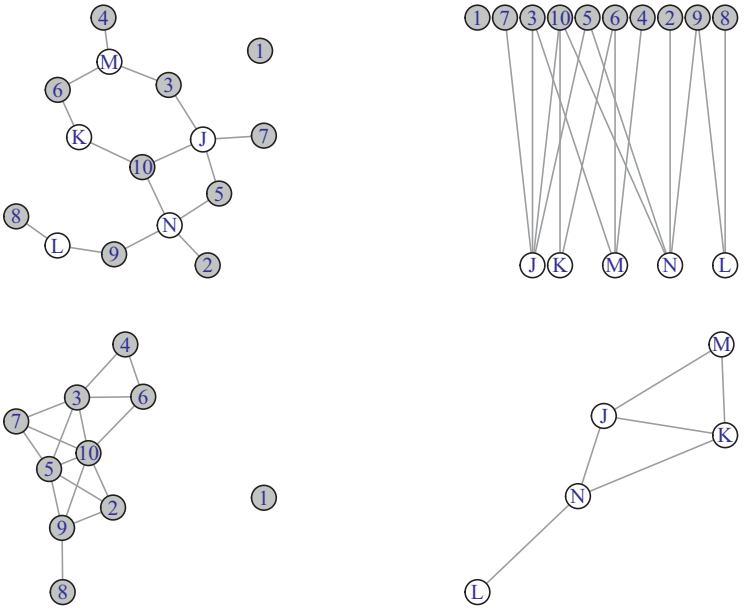

Figure 1.9 The top two networks show a single bipartite network plotted in two different ways. The bottom two networks are the results of projecting the top network onto each of the two node types.

Table 1.8 Bipartite adjacency matrix with two node types.

	J	K	L	M	N
1	0	0	0	0	0
2	0	0	0	0	1
3	1	0	0	1	0
4	0	0	0	1	0
5	1	0	0	0	1
6	0	1	0	1	0
7	1	0	0	0	0
8	0	0	1	0	0
9	0	0	1	0	1
10	1	1	0	0	1

edge relationships. These relationships can then be separated or combined to investigate new networks with novel properties.

For example, in social networks, edges could represent work, family, *and* local community relationships. In language networks, edges could represent phonological (sound) *and* semantic (meaning) relationships. Multiplex networks combine these edge types in different ways to understand their relative contributions and patterns of interaction. Figure 1.10 shows a full network with three different edge types. It also shows the network associated with each edge type separately.

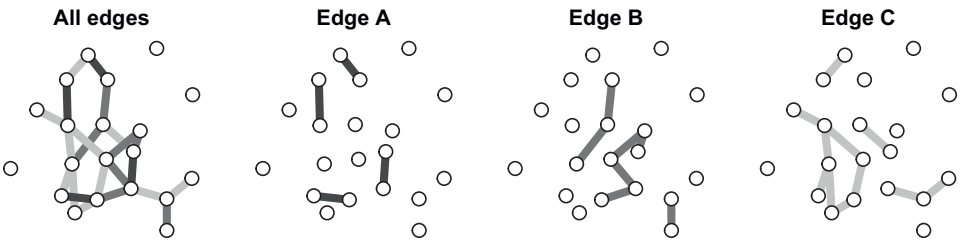

Figure 1.10 Multiplex network in which the combined representation is made up of edges of three different types.

1.6 Turning Data into Networks

Data does not always come to us in the form of an edge list or adjacency matrix. Instead, there are surveys, lists of remembered stimuli, decisions made by individuals at different times, natural language, collections of documents, and so on. All of these can be turned into networks, but there is no single answer as to how to do it.

I will offer several examples here, but many others are provided throughout the book and many more are published each day. In general, however, the workflow is as follows:

First, identify the nodes. Nodes are typically based on theoretical questions: Are we interested in individuals, groups of individuals, specific behaviors, categories of behavior, or something else? This is a research decision, but networks invite us to consider it carefully. For example, we might want to know how disease symptoms progress through a population. We could treat individuals as the nodes and attempt to identify edges between individuals based on symptom similarity. Alternatively, we could treat symptoms as nodes, noting that insomnia is comorbid with depression and possibly has a directed edge, such that one leads to the other. The breadth of possibilities is wide and it is useful to consider them with some creativity.

Second, identify the edges. When the data comes in the form of relationships between nodes of interest, things are often clear. In many cases, though, there are yet more veiled relationships. They might be inferred from the data or we might bring some outside data to bear on the problem. For example, when nodes have collections of features they share with other nodes, we have a bipartite network. We could choose a projection from a bipartite network to form edges. But edges could also be weighted based on the number of shared features (as in bibliometric networks, e.g., Newman, 2018), or on the cosine similarity between vectors of shared neighbors (as in semantic space models, see Chapter 12), or on other measures of similarity or distinctiveness (see Chapter 8). We could have many people evaluate the similarities among pairs (see Chapter 13) or use statistical models to compute similarities among documents or words that appear otherwise unrelated. When order matters, we can use ordered lists to construct networks that would regenerate the lists that produced them (Zemla & Austerweil, 2018). It is often useful to consider different edge possibilities as representing different hypotheses about what matters. Then we can see how well they explain the phenomenon of interest.

1.6.1 Survey Data

If we have a list of items that we care about, such as long-term health risks, we can ask a group of people to evaluate each of these along a common scale. We then have rows of individuals and columns that indicate how they answered each question. **Network psychometrics** approaches have developed numerous tools for asking about the relationships among such variables (e.g., Christensen et al., 2018; Isvoranu et al., 2022). The general principle is that one infers edges based on the correlations or covariance between questions (i.e., nodes) in the pattern of people's responses to them. Edges with more weight represent more similar patterns of responses across participants. Now nodes are not participants but the items they are evaluating. Edges are similarities in evaluation.

One of the key complications of these approaches is that the number of edges grows dramatically with the number of variables. If there were 20 variables (nodes), then we would have 190 edges to estimate. And we may not have enough data to do that well. To deal with this problem, filtering methods can be used to reduce the number of edges. Several of these filtering methods are shown in Figure 1.11. The development of these methods is ongoing. They are also not without controversy. That said, there is no better way to understand the controversy than to see it in action with your own data.

Suppose we have binary responses to survey data as shown in Table 1.9. This data might represent "yes" or "no" responses to a series of five questions each answered by an individual participant.

Table 1.9 A sample of cross-sectional survey data. Rows are participants and columns are binary responses to survey questions.

	A	B	C	D	E
1	0	0	0	0	0
2	0	1	0	1	1
3	0	0	0	0	0
4	1	1	0	0	0
5	0	1	0	1	0

Table 1.10 A sample of cross-sectional survey data. Rows are participants and columns are responses to survey questions with continuous ratings on a scale from 1 to 7.

	A	B	C	D	E	F	G
1	4	4	4	5	5	5	5
2	3	3	5	3	1	5	4
3	6	4	4	6	5	5	5
4	2	2	4	3	3	5	4
5	6	5	3	6	3	4	6

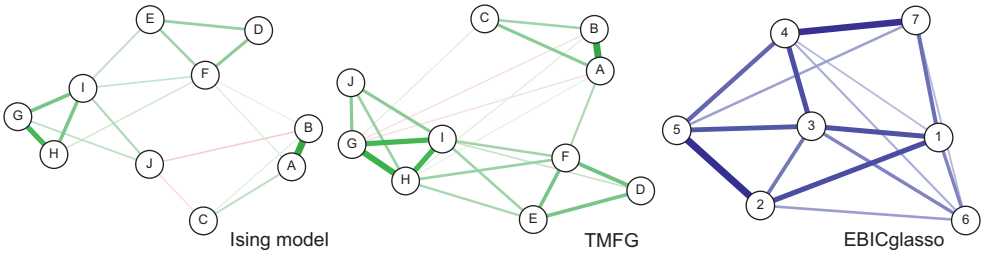

Figure 1.11 Network psychometric graphs based on survey measures. Networks shown are the output of the Ising model and the TMFG information filtering graph for a 10 question survey with 1,700 individuals. Also shown is the output of an EBIC graphical lasso model from a stylized data set representing continuous survey data from 150 participants on 7 survey questions, each rated from 1 to 7.

One method for identifying edges in this data is the *Ising model* (Van Borkulo et al., 2014). This is based on logistic regression, predicting each variable based on the other variables – giving stronger edges to better predictors. It then penalizes edge weights by limiting the total edge weight possible over the entire graph, shrinking smaller edge weights toward zero. The Ising model attempts to identify the best solution by choosing the network that makes the best trade-off between minimizing error and minimizing model complexity. This trade-off is handled using a variation of the Bayesian Information Criterion, which is discussed in more detail in Chapter 6. Figure 1.11 shows the output of this model run on a binary data set with 12 variables and 3,000 individuals.

Information filtering networks offer another approach. This approach computes edges based on the binary equivalent of Pearson's correlations. These edges are then ordered and added to the network one at a time according to a set of rules that minimizes edge overlap (see Massara et al., 2017). This produces the TMFG (triangulated maximally filtered graph) network shown in Figure 1.11. See Christensen et al. (2018) for further details on this approach and the Ising model.

If the data is continuous, such as survey data with rating scales, we have something like that shown in Table 1.10. Edges can be created here using the partial correlation coefficients combined with what is called a *graphical lasso*. The graphical lasso penalizes edges, similar to the Ising model mentioned earlier, simplifying the overall edge structure by keeping only those edges with the strongest weights (Epskamp & Fried, 2018). This produces the EBICglasso network shown in Figure 1.11.

There are many other approaches. Rarely do a few months go by that someone does not come up with a new and insightful way to generate nodes and edges. The possibilities are far from exhausted.

In this chapter, we have gone from thinking about nodes and edges to building a variety of different network types from our data. In the next chapter, we will start evaluating network properties.