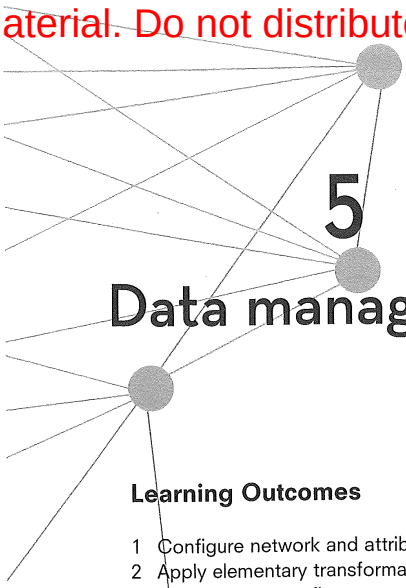


5 Data management

Learning Outcomes

1 Configure network and attribute data for standard software packages
2 Apply elementary transformations to matrix data
3 Extract and reconfigure network and attribute data

5.1 Introduction

In this chapter we discuss how to format network data for import into a network analysis software package, how to transform network data to make it suitable for different analyses, and how to export network data and results for use in other programs, such as statistical packages. For obvious reasons, this chapter is a little more specific to the UCINET program than most chapters of the book. The data import section discusses the various choices available for formatting network data in electronic files. We also make suggestions for checking and cleaning newly imported data. The section on data transformation focuses on common adjustments made to network data, such as imputing missing values, symmetrizing, dichotomizing, aggregating and subsetting network data. We do not discuss converting two-mode data to one-mode data as we devote a separate chapter to that topic (see Chapter 13). The data export section discusses ways of transferring data and results to statistical programs as well as to spreadsheet and word-processing software.

5.2 Data import

One of the most important steps in any network analysis is formatting the data for import into a network analysis software package. Familiarizing yourself with

standard data formats *before* entering or downloading data can save a great deal of time – it can be very costly to have to reformat data later on.

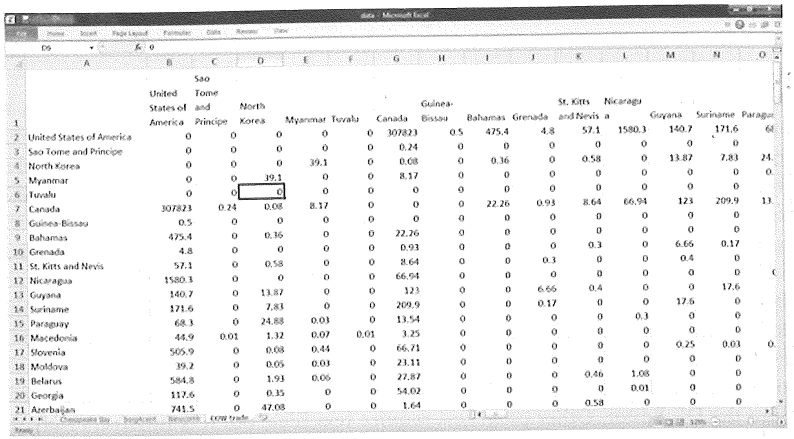
Regardless of how data is obtained, eventually it must be held in an electronic file, such as a spreadsheet, database, or text file. For large datasets, a proper database such as Microsoft Access or MySQL is useful, but since few network analysis programs can read database files directly, they entail an extra step of converting to something the programs can read. For most users, we recommend using Microsoft Excel as a sort of universal translator. Current versions of Excel can handle large datasets, and the fact that each datum is stored in its own spreadsheet cell solves the parsing issues that arise when programs have to read text files, such as having to decide whether 'Pitney Bowes' refers to two nodes or one, and whether '7,100' represents seven thousand one hundred, seven and one tenth, or a pair of numbers seven and one hundred. In the case of UCINET, Excel files can be imported directly or simply cut and pasted into the DL Editor. Other programs require text files, but it is relatively easy to convert Excel files to text as needed. Several network analysis programs read text files written in the DL (Data Language) data entry language. We give some examples later in this chapter.

The issue of which type of file to use for storing data is not as important as the format of the data. The UCINET program can read a wide variety of formats (such as matrix formats and list formats), allowing the user to choose one that is convenient for their situation (e.g., typed in from paper surveys, saved from electronic surveys, downloaded from archival sources). We begin by discussing matrix formats.

5.2.1 Matrix formats

A conceptually straightforward data format is the full matrix format. Figure 5.1 shows a screenshot of a network in full matrix format held in a Microsoft Excel file. The example shows a standard one-mode dataset, meaning that the rows and columns are the same. Note that the first row and column are used for node labels, and these particular labels happen to contain spaces. It is also possible to use numeric labels, or no labels at all (in which case nodes will be assigned the ordinal numbers from 1 to N as labels). Note also that these matrix entries are tie strengths, meaning there are values other than simple 1s and 0s. The matrix can be easily transferred to UCINET by cutting and pasting into the UCINET DL Editor. A screenshot of the DL Editor is shown in Figure 5.2, using a different dataset.

In general, matrix formats are best used with dense networks. If the network is very sparse, as most networks are, the adjacency matrix will consist mostly of zeros, in which case it is more economical to use a format where non-ties are not entered, such as the list formats. Many-valued datasets, such as one giving the



	United States of America	Sao Tome and Principe	North Korea	Myanmar	Tuvalu	Canada	Guinea-Bissau	Bahamas	Grenada	St. Kitts and Nevis	Nicaragua	Guyana	Suriname	Paraguay
United States of America	0	0	0	0	0	307823	0.5	475.4	4.8	57.1	1580.3	140.7	171.6	68
Sao Tome and Principe	0	0	0	0	0	0.24	0	0	0	0	0	0	0	
North Korea	0	0	0	39.1	0	0.08	0	0.36	0	0.58	0	13.87	7.83	24
Myanmar	0	0	39.1	0	0	8.17	0	0	0	0	0	0	0	0
Tuvalu	0	0	0	0	0	0	0	0	0	0	0	0	0	
Canada	307823	0.24	0.08	8.17	0	0	0	22.26	0.93	8.64	66.94	123	209.9	13
Guinea-Bissau	0.5	0	0	0	0	0	0	0	0	0	0	0	0	
Bahamas	475.4	0	0.36	0	0	22.26	0	0	0	0	0	0	0	
Grenada	4.8	0	0	0	0	0.93	0	0	0	0.3	0	6.66	0.17	
St. Kitts and Nevis	57.1	0	0.58	0	0	8.64	0	0	0.3	0	0	0.4	0	
Nicaragua	1580.3	0	0	0	0	66.94	0	0	0	0	0	0	0	
Guyana	140.7	0	13.87	0	0	123	0	0	6.66	0.4	0	0	17.6	
Suriname	171.6	0	7.83	0	0	209.9	0	0	0.17	0	0	17.6	0	
Paraguay	68.3	0	24.88	0.03	0	13.54	0	0	0	0	0.3	0	0	
Macedonia	44.9	0.01	1.32	0.07	0.01	3.25	0	0	0	0	0	0	0	
Slovenia	505.9	0	0.08	0.44	0	66.71	0	0	0	0	0	0.25	0.03	0
Moldova	39.2	0	0.05	0.03	0	23.11	0	0	0	0	0	0	0	
Belarus	584.8	0	1.93	0.06	0	27.87	0	0	0	0.46	1.08	0	0	
Georgia	117.6	0	0.35	0	0	54.02	0	0	0	0	0.01	0	0	
Azerbaijan	741.5	0	47.08	0	0	1.64	0	0	0	0.58	0	0	0	

Figure 5.1 Full matrix format.

physical distances between pairs of nodes, are completely dense, since there is a value for every pair of nodes. For these kinds of data, matrix formats do a good job.

The matrix format is also useful for importing attribute data, where each column in the matrix is an attribute. For example, Figure 5.2 shows a screenshot of the UCINET DL Editor with four attributes pertaining to a network of five nodes. Unfortunately, most network analysis programs require attributes to have numeric values, so a codebook must be kept in order to know whether a '1' under

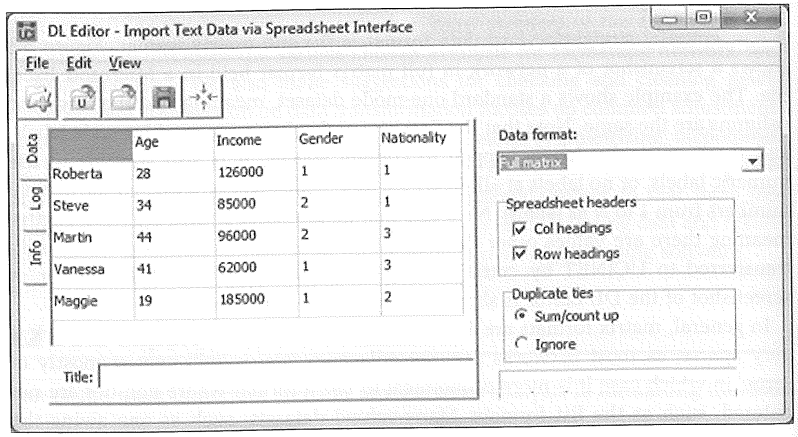


Figure 5.2 Attribute data formatted as a full matrix in UCINET's DL Editor.

'gender' means male or female.[1] We include a fuller discussion of working with attributes later in the chapter.

5.2.2 List formats

When the number of links that actually exist in a network is much smaller than the number that could exist, the most economical thing to do is store only the ties and leave out the non-ties. There are a number of formats that do this, including the nodelist and edgelist formats.

Nodelists

The nodelist format is the simplest and most economical of all the formats. It is used only for binary data (i.e., presence/absence of ties; no tie strengths). Figure 5.3 shows a screenshot of the UCINET DL Editor spreadsheet with data in a nodelist format representing a six-node undirected network. The first name in each row gives the node that is 'sending' a tie – the ego. The names that follow in the same row are the nodes receiving each tie – the alters. Hence, the first row with 'Bill Smith', 'Carrie Jones', 'Doug Johnson' and 'Eric Morrison' states that there is a tie from 'Bill Smith' to 'Carrie Jones', and from 'Bill Smith' to 'Doug Johnson', and so on. In addition, because the 'force symmetry' option is on, the program automatically supplies a tie from 'Carrie Jones' to 'Bill Smith', from 'Doug Johnson'

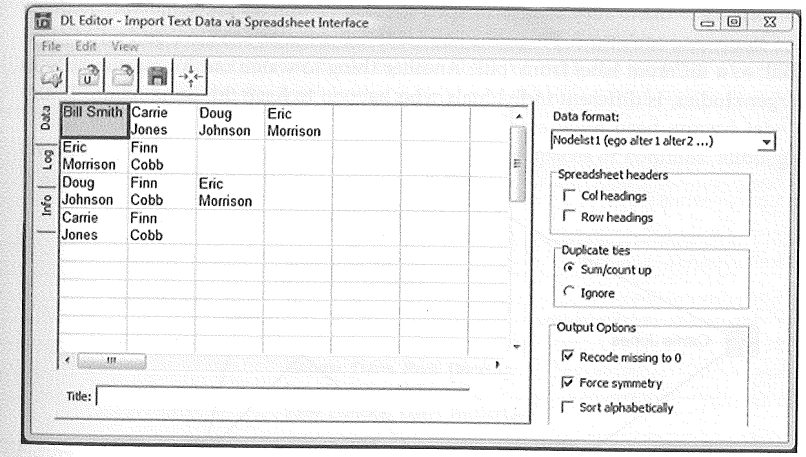


Figure 5.3 Nodelist data.

[1] An exception is NetDraw, which allows text values in variables.

	Bill Smith	Carrie Jones	Doug Johnson	Eric Morrison	Finn Cobb
Bill Smith	0	1	1	1	0
Carrie Jones	1	0	0	0	1
Doug Johnson	1	0	0	1	1
Eric Morrison	1	0	1	0	1
Finn Cobb	0	1	1	1	0

Matrix 5.1 Matrix generated from a nodelist.

to 'Bill Smith', and so on for all the others. From this data, the program constructs an adjacency matrix (see Matrix 5.1). The order of rows and columns in the adjacency matrix is determined by the order in which the program encounters the names of the nodes. (This order is overridden if 'sort alphabetically' is checked.) A visualization of this network is given in Figure 5.4.

In any list format, the node identifiers can be numbers instead of names. Using numbers can be more economical than typing long names over and over again, and the labels for the nodes can be added later. One thing to note, if using labels directly, is to make sure always to spell the names exactly the same way and use the same case each time. Programs like UCINET are case-sensitive and will regard 'Bill' as a different label from 'bill'. Another thing to watch out for, particularly in larger studies, is different individuals who happen to have the same name.

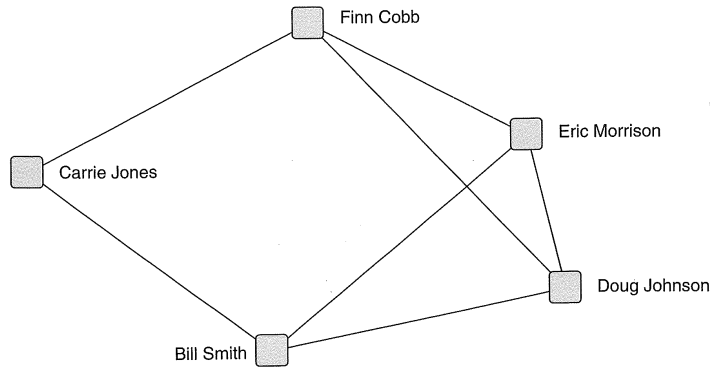


Figure 5.4 Network constructed using nodelist format.

Figure 5.5 shows the same data as Figure 5.3 placed in a text file using the DL data description language mentioned at the beginning of the chapter. The 'dl' at the beginning tells the software to expect the DL syntax. The 'n=6' gives the number of actors, while 'format=nodelist1' tells the program to expect a one-mode nodelist format, and 'symmetric=yes' states that this is symmetric data, so that every link should be treated as reciprocated, and 'data:' indicates the end of information about the data and the beginning of the data itself. Note that node labels are surrounded by full quotation marks. If this were not done, the program reading the data would interpret Bill as one node and Smith as another.

```
dl n=6
format=nodelist1
symmetric=yes
labels embedded
data:
"Bill Smith" "Carrie Jones" "Doug Johnson" "Eric Morrison"
"Eric Morrison" "Finn Cobb"
"Doug Johnson" "Finn Cobb" "Eric Morrison"
"Carrie Jones" "Finn Cobb"
```

Figure 5.5 Data from Figure 5.3 in a text file using the DL data description language.

When dealing with two-mode data, we need to indicate both the number of rows and columns. For example, the data file in Figure 5.6 describes a person-by-activity matrix. This time we have to take stock of both the number of actors ('nr=3'), which will be the rows in the matrix, and the number of events ('nc=5'), which will be the columns in the matrix. No quotes were used around labels because none contained any punctuation marks like spaces or commas, although it never hurts to include the quotes.

```
dl nr=3, nc=5
format = nodelist2
row labels embedded
column labels embedded
data:
George, Darts, Pool, Dancing
Sue, Dancing, Volleyball
Sally, Dancing, Darts, Basketball
```

Figure 5.6 Two-mode nodelist data in a text file using the DL syntax.

Edgelists

The edgelist format consists of a set of rows in which each row represents a tie in the network. Each row has two columns indicating the pair of nodes that

have the tie. Optionally, a third column can be included which gives the strength of the tie. When network data is stored in databases, it is frequently organized in this way (recall the IMDb example in Chapter 4). Figure 5.7 shows a screenshot of the UCINET DL Editor spreadsheet with data in a simple edgelist format representing a six-node undirected network. This type of format is also applicable to two-mode data (in which case it is referred to as edgelist2).

Most network studies collect multiple relations on the same set of nodes. For example, a researcher might ask respondents about their friendships, their professional relationships, family ties, and so on. One way of entering

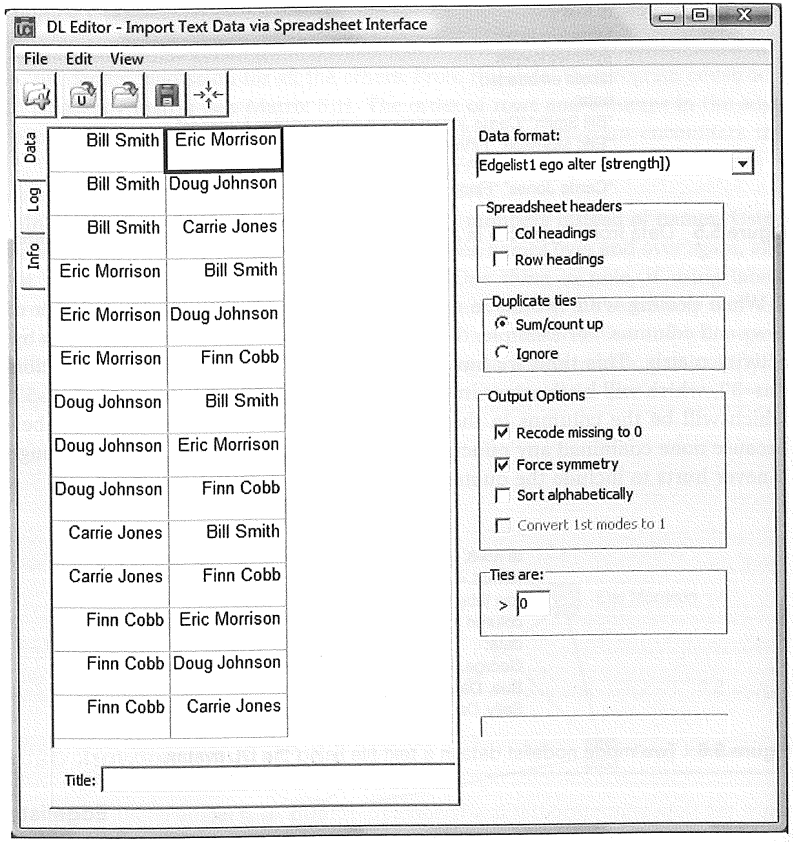


Figure 5.7 Edgelist format for the same data as in Figure 5.3.

multirelational data is using a variation on the edgelist called the edgelist23 format. This consists of node–node–relation triples, as shown in Figure 5.8. The data is from UCINET's Padgett dataset, which contains two different kinds of social ties, marriage and business.

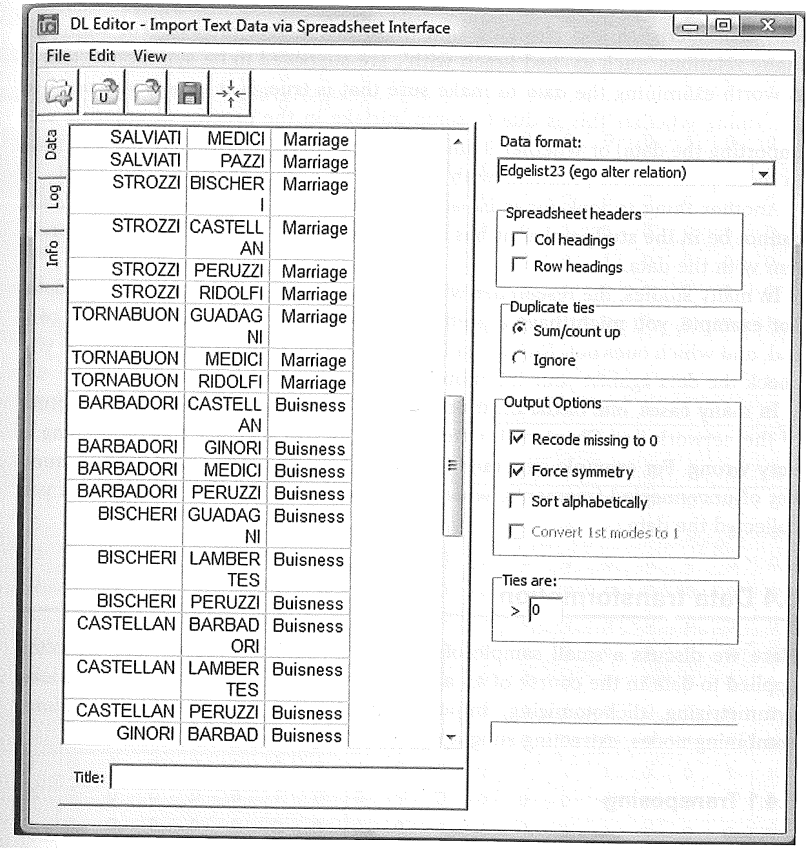


Figure 5.8 Padgett data in edgelist23 format.

5.3 Cleaning network data

Once the data is imported, it is advisable to examine it in some detail. There are usually a number of problems, and if these are detected early it can save a lot of time in repeating analyses done on the flawed data. One kind of problem to look

for is repeated nodes. This occurs when an actor has been entered twice, or more commonly when there are slight differences in how the node's name was typed. Another common problem is that there are some missing actors. This could be because of non-response or an error in entering or copying the data. The non-response issue is a serious problem to which we return later.

It is always worth thinking about whether the data should logically have certain characteristics and checking that those are in fact present. For example, many relations, such as 'had lunch with', are supposed to be symmetric, and it is worth examining the data to make sure that is true. If it is not, you need to determine whether this is due to some mistake in the process of entering and importing the data, or whether it simply reflects what the respondents said (due to recall problems or problems with the way the questionnaire was worded).

Another thing to look for is isolates. In some data collection designs a node cannot be in the study unless it has ties, so any isolates you find suggest a problem with the data.

In many studies, the researcher will have some ethnographic feel for the data. For example, you might have a pretty good idea of which nodes should be central, and which ones not. Running a quick centrality analysis early on will let you check the data against your intuition.

In many cases, one of the most useful things you can do is construct a picture of the network (see Chapter 7). Often you can tell at a glance that something is very wrong. For example, you might see that the network is divided into a number of unconnected fragments, which might not make any sense given how you collected the data.

5.4 Data transformation

Here we discuss a small sample of the myriad transformations that are often applied to data in the course of an analysis. These include transposing matrices, symmetrizing, dichotomizing, imputing missing values, combining relations, combining nodes, extracting subgraphs, and many more.

5.4.1 Transposing

To transpose a matrix is to interchange its rows with its columns (see Figure 5.9). When applied to a non-symmetric adjacency matrix, this has the effect of reversing the direction of all the arcs (see Figure 5.10). This can be helpful in maintaining a consistent interpretation of the ties in a network. When we construct adjacency matrices from surveys, we generally do it in such a way that the rows correspond to respondents (egos) and the columns correspond to the people mentioned by ego (alters). However, it is also convenient to think of the row node as sending to the column node, as in sending information or resources. These two conventions can be in conflict. For example, suppose the survey asks

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
A	0	1	1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0
B	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
C	0	1	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
D	0	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
E	0	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
F	0	1	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0
G	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
H	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0
I	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0
J	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0	0	0
K	0	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	0
L	0	0	0	0	0	0	0	1	0	1	0	0	1	0	0	0	0	0
M	0	0	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0
N	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	0
O	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1
P	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	1
Q	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	1
R	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0

(a)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
A	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	1	0	1	1	1	1	0	0	0	0	0	0	0	0	1	0	0	0
C	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
D	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
E	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
F	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
H	0	0	1	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
I	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	1	0	0
J	0	0	0	0	0	0	0	1	0	0	0	1	1	1	0	0	1	1
K	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0
L	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0
M	0	0	0	0	0	0	0	1	0	1	0	1	0	1	1	0	0	0
N	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
O	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
P	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1
Q	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0

(b)

Figure 5.9 A matrix and its transpose: (a) who likes whom; (b) who is liked by whom.

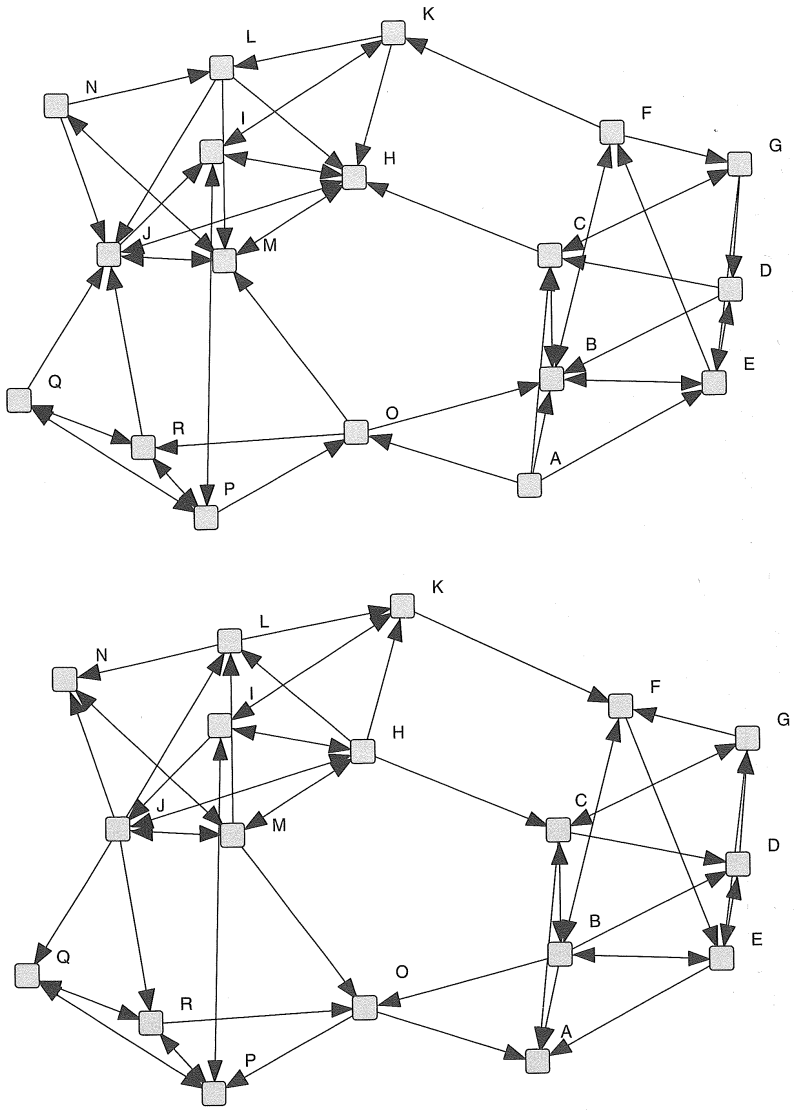


Figure 5.10 Transposing an adjacency matrix is equivalent to reversing the direction of the arrows.

'who do you seek advice from?'. A '1' in cell (3, 7) means that person 3 says they seek advice from person 7. But in which direction does advice flow? Advice is flowing from person 7 to person 3. In this case, it might be useful to transpose the matrix and think of it as who gives advice to whom. A similar situation occurs with food webs, where we have data on which species eat which other species. Ecologists like to reverse the direction of the arrows because they think in terms of the direction of energy flow through the ecosystem (e.g., carbon flowing from the prey to the predator).

So far we have only considered transposing two-dimensional matrices. Stacked datasets can be seen as three-dimensional matrices consisting of rows, columns and layers or slices. In these matrices, three different transpositions can be done: interchanging rows with columns, rows with layers, and columns with layers. The column–layer transposition is particularly useful. Suppose, for example, we have run a centrality analysis on UCINET's Padgett dataset, which we encountered earlier. This dataset contains two matrices corresponding to marriage and business ties among Florentine families during the Renaissance. When we run centrality on it, the result is a new three-dimensional dataset consisting of two person-by-centrality measure layers, one for each type of tie, as shown in Matrix 5.2. This way of organizing the results lends itself to answering a question like 'how does the Medici's centrality vary depending how we measure centrality?' but is not as helpful for answering 'how does the Medici's centrality in the marriage network compare with their centrality in the business network?'. For that, we would rather have the results for the two networks side by side and have the measures correspond to the different matrices, as shown in Matrix 5.3. We can accomplish this transposition easily in UCINET's Matrix Algebra command-line facility. Assuming the centrality results are in a dataset called PADGETT-CENT, we would type:

```
-->display transpose(padgett-cent col layer)
```

5.4.2 Imputing missing data

Missing data can be a problem in full network research designs. The most common kind of missing data is where a respondent has chosen not to fill out the survey. This creates a row of missing values in the network adjacency matrix. For some analyses, such as the QAP regressions that are discussed in Chapter 8, this is not a big problem. Many graph-theoretic procedures, however, such as centrality measures, will treat the missing values as non-ties, which is simply incorrect.

An obvious solution is to eliminate that node from the analysis altogether (deleting both their row and their corresponding column in the adjacency matrix). The trouble with this is that the remaining network is a little misleading. If the missing

Marriage	Degree	BonPwr	2Step	ARD	Eigen	Bet
ACCIAIUOL	1.0	301.3	6.0	5.9	0.1	0.0
ALBIZZI	3.0	557.4	10.0	7.8	0.2	19.3
BARBADORI	2.0	482.7	9.0	7.1	0.2	8.5
BISCHERI	3.0	644.6	8.0	7.2	0.3	9.5
CASTELLAN	3.0	590.4	6.0	6.9	0.3	5.0
GINORI	1.0	171.3	3.0	5.3	0.1	0.0
GUADAGNI	4.0	660.3	9.0	8.1	0.3	23.2
LAMBERTES	1.0	202.8	4.0	5.4	0.1	0.0
MEDICI	6.0	982.7	11.0	9.5	0.4	47.5
PAZZI	1.0	103.0	2.0	4.8	0.0	0.0
PERUZZI	3.0	628.3	6.0	6.8	0.3	2.0
PUCCI	0.0	0.0	0.0	0.0	0.0	0.0
RIDOLFI	3.0	778.2	11.0	8.0	0.3	10.3
SALVIATI	2.0	333.8	7.0	6.6	0.1	13.0
STROZZI	4.0	811.2	8.0	7.8	0.4	9.3
TORNABUON	3.0	742.9	10.0	7.8	0.3	8.3

Business	Degree	BonPwr	2Step	ARD	Eigen	Bet
ACCIAIUOL	0.0	0.0	0.0	0.0	0.0	0.0
ALBIZZI	0.0	0.0	0.0	0.0	0.0	0.0
BARBADORI	4.0	756.2	9.0	6.8	0.4	25.0
BISCHERI	3.0	663.5	5.0	5.4	0.3	2.5
CASTELLAN	3.0	754.5	7.0	6.0	0.4	5.0
GINORI	2.0	370.7	7.0	5.4	0.2	0.0
GUADAGNI	2.0	452.9	4.0	4.4	0.2	0.0
LAMBERTES	4.0	838.5	5.0	5.9	0.4	6.0
MEDICI	5.0	471.8	7.0	6.9	0.2	24.0
PAZZI	1.0	142.7	5.0	4.4	0.1	0.0
PERUZZI	4.0	908.5	7.0	6.5	0.5	13.5
PUCCI	0.0	0.0	0.0	0.0	0.0	0.0
RIDOLFI	0.0	0.0	0.0	0.0	0.0	0.0
SALVIATI	1.0	142.7	5.0	4.4	0.1	0.0
STROZZI	0.0	0.0	0.0	0.0	0.0	0.0
TORNABUON	1.0	142.7	5.0	4.4	0.1	0.0

Matrix 5.2 Three-dimensional matrix containing centrality measures computed on the Padgett dataset.

Degree	Marriage	Business
ACCIAIUOL	1	0
ALBIZZI	3	0
BARBADORI	2	4
BISCHERI	3	3
CASTELLAN	3	3
GINORI	1	2
GUADAGNI	4	2
LAMBERTES	1	4
MEDICI	6	5
PAZZI	1	1
PERUZZI	3	4
PUCCI	0	0
RIDOLFI	3	0
SALVIATI	2	1
STROZZI	4	0
TORNABUON	3	1

BonPwr	Marriage	Business
ACCIAIUOL	301.3	0.0
ALBIZZI	557.4	0.0
BARBADORI	482.7	756.2
BISCHERI	644.6	663.5
CASTELLAN	590.4	754.5
GINORI	171.3	370.7
GUADAGNI	660.3	452.9
LAMBERTES	202.8	838.5
MEDICI	982.7	471.8
PAZZI	103.0	142.7
PERUZZI	628.3	908.5
PUCCI	0.0	0.0
RIDOLFI	778.2	0.0
SALVIATI	333.8	142.7
STROZZI	811.2	0.0
TORNABUON	742.9	142.7

2-Step	Marriage	Business
ACCIAIUOL	6	0
ALBIZZI	10	0
BARBADORI	9	9
BISCHERI	8	5
CASTELLAN	6	7
GINORI	3	7
GUADAGNI	9	4
LAMBERTES	4	5
MEDICI	11	7
PAZZI	2	5
PERUZZI	6	7
PUCCI	0	0
RIDOLFI	11	0
SALVIATI	7	5
STROZZI	8	0
TORNABUON	10	5

Matrix 5.3 Centrality measures after transposing columns and layers of Matrix 5.2 (some measures omitted for brevity).

node is the most important in the network – and it is not unusual for important people not to have time to fill out the survey – the picture we get will be very different from what we should have had. Moreover, since the other nodes made responses about that node, removing them means wasting a lot of good data. It would seem worthwhile, then, to search for ways to retain the problematic node.

In the case of symmetric or undirected relations, a simple cure is to fill in any missing rows with the data found in the corresponding column. The assumption is that, if the respondent had been able to answer, they would have listed all the actors that mentioned them. This may not be exactly right, but it will be more accurate than treating the missing values as zeros.[2] UCINET has a command called REPLACENA within Matrix Algebra to do this. For example, given a matrix called MARRIAGE with a couple of rows of missing data, you would type:

```
-->cleanedmarriage = replacena(marriage transpose(marriage))
```

This tells the program to construct a transposed version of the MARRIAGE matrix (in which the columns become the rows and vice versa), then replace all missing values in the MARRIAGE matrix with the corresponding value in the transposed matrix. This effectively replaces the missing rows in MARRIAGE with the corresponding columns, yielding a new file called 'cleanedmarriage'.

For non-symmetric relations, such as 'seeks advice from', this technique would not make sense. However, if you were wise enough to have asked your respondents both 'who do you seek advice from?' and 'who seeks advice from you?', you can use the transpose of the second matrix to fill in the missing rows in the first, and vice versa. In other words, you assume that if someone says Bill seeks advice from them, then if Bill had been able to answer the survey, he would have said he seeks advice from that person.

For more sophisticated ways of imputing missing values, the reader is advised to consult the relevant literature. For example, Butts (2003) presents a Bayesian approach intended for the case of cognitive social structure data. For general matrices, Candès and Recht (2012) present ingenious methods for recovering missing cells. In general, it is a good idea to run the analyses with different ways of handling missing values, including simply removing nodes with missing data, to see if the results are robust. If they are not, there is the danger that any findings are an artifact of the method used to handle missing values.

5.4.3 Symmetrizing

Symmetrizing refers to creating a new dataset in which all ties are reciprocated (and perhaps regarded as undirected). There are many reasons to symmetrize

[2] Of course, one should not do this when calculating reciprocity rates, as it will artificially inflate those values.

data. One very practical reason is that some analytical techniques, such as multidimensional scaling, assume symmetric data. In other cases, symmetrizing is part of data cleaning. For example, when we ask respondents to name their friends using an open-ended questionnaire item, we often find unintended asymmetry because respondents simply forget to mention people, as noted in the previous chapter. In these cases, we often create a new, symmetric adjacency, using the rule that if either person mentioned the other, then there is a tie. We call this the OR, or union, rule. Alternatively, if we suspect name-dropping, we might adopt a stricter rule, namely that only if both people mention each other will we consider it a tie. This is called the AND, or intersection, rule. Obviously, the union rule creates networks denser than the original, while the intersection rule makes them sparser. As discussed in Chapter 14, the latter can be helpful when dealing with large networks.

In other cases, we symmetrize in order to study an underlying symmetric relationship that is not quite the same thing as the observed ties. For example, suppose we have asked respondents from whom do they receive advice. In seeking advice, we know that an actor reveals the problem they are trying to solve. In this sense, there is an exchange of information. It may be that this is the social relation we are really interested in studying, perhaps because we see it as a proxy for a certain level of collaboration or intimacy. In this case, we symmetrize using the rule that if either gives advice to the other, we say there is an exchange tie.

From the point of view of a matrix representing a network, when we symmetrize we are comparing an (i, j) entry with the corresponding (j, i) entry and, if needed, making them the same. The union rule corresponds to taking the larger of the two entries. The intersection rule takes the smaller of the two. Many other options are possible as well. For valued data we might consider taking the average of the two entries. For example, if i estimates having had lunch with j eight times in a month, but j estimates having lunched with i ten times, we can view these as two measurements of the same underlying quantity, and use the average as the best estimate of that quantity.

5.4.4 Dichotomizing

Another common data transformation is dichotomizing, which refers to converting valued data to binary data. In this case, we take a valued adjacency matrix and set all cells with a value greater than (or less than, or exactly equal to) a certain threshold to 1, and set all the remaining cells to 0 (see Matrices 5.4 and 5.5). The usual reason for doing this is again very practical: some methods, especially graph-theoretic methods, are only applicable to binary data. Also, dichotomizing with a high cut-off can serve to reduce the density of the network, which is useful in handling large networks (see Chapter 14).

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0	2	1	0	2	0	0	2	2	2	2	1	1	1	0
2	2	0	3	3	1	4	2	0	2	1	1	2	0	2	0
3	1	3	0	6	1	2	2	1	2	0	2	2	1	1	0
4	0	3	6	0	2	2	1	0	0	0	4	3	1	0	0
5	2	1	1	2	0	1	1	2	1	1	2	1	1	0	0
6	0	4	2	2	1	0	1	2	2	0	2	0	1	0	0
7	0	2	2	1	1	1	0	1	1	0	1	0	2	1	0
8	2	0	1	0	2	2	1	0	2	1	2	0	2	0	0
9	2	2	2	0	1	2	1	2	0	3	3	0	1	1	0
10	2	1	0	0	1	0	0	1	3	0	3	1	0	1	0
11	2	1	2	4	2	2	1	2	3	3	0	0	1	0	0
12	1	2	2	3	1	0	0	0	0	1	0	0	0	0	0
13	1	0	1	1	1	1	2	2	1	0	1	0	0	1	0
14	1	2	1	0	0	0	1	0	1	1	0	0	1	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Matrix 5.4 Original valued data.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0	1	0	0	1	0	0	1	1	1	1	0	0	0	0
2	1	0	1	1	0	1	1	0	1	0	0	1	0	1	0
3	0	1	0	1	0	1	1	0	1	0	1	1	0	0	0
4	0	1	1	0	1	1	0	0	0	0	1	1	0	0	0
5	1	0	0	1	0	0	0	1	0	0	1	0	0	0	0
6	0	1	1	1	0	0	0	1	1	0	1	0	0	0	0
7	0	1	1	0	0	0	0	0	0	0	0	0	1	0	0
8	1	0	0	0	1	1	0	0	1	0	1	0	1	0	0
9	1	1	1	0	0	1	0	1	0	1	1	0	0	0	0
10	1	0	0	0	0	0	0	0	1	0	1	0	0	0	0
11	1	0	1	1	1	1	0	1	1	1	0	0	0	0	0
12	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
14	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Matrix 5.5 Dichotomized data.

When dichotomizing for these very practical reasons it is usually advisable to dichotomize at different levels and run the analyses on each of the resulting datasets. This approach retains the richness of the data and can reveal insights into the network structure that would not be easy to deduce from techniques designed to deal with valued data directly. It also gives you an idea of the extent

to which your findings are robust across different definitions of ties. Unless theoretically motivated, you do not want results that hinge on a particular, perhaps arbitrary, choice of dichotomization thresholds.[3]

In other cases we dichotomize because different research questions call for different kinds of ties. For example, we might have measured ties using a scale like this: 0 = don't know, 1 = acquaintance, 2 = friend, 3 = best friend. With respect to estimating the chance of hearing some random bit of information, perhaps we would dichotomize at anything greater than 0. But with respect to estimating a person's feeling of having emotional support, we might count only friendships, and dichotomize at larger than 1.

5.4.5 Combining relations

As noted in the discussion of edgelist23 data formats, most network studies collect multiple relations on the same set of nodes. For some analyses, however, we will combine some of these separate relations into one. For example, we might take three separate network questions, such as 'who do you attend sports events with?', 'who do you go to the theatre with?', and 'who do you go out to dinner with?' and combine them into a more general, analytically defined, relation, such as 'who socialized with whom'. More broadly, we might take several relations involving friendship, support, liking and so on and combine them to create a category of relations that we might call 'expressive ties'. Similarly, we might take a number of network questions about coordinating at work, getting work advice from, and so on, and build an instrumental tie matrix.

To actually construct the adjacency matrix for an aggregated relation we can use tools like UCINET's Boolean Combination procedure, or simply sum the individual adjacency matrices. For example, in UCINET we might use the Matrix Algebra procedure to add a set of matrices corresponding to positive relations. The following command creates a new relation called 'positive' from all the relations in the Sampson dataset (Sampson 1969) that we deem to be positive:

```
-->positive = add(samplk1 samplk2 samplk3 sampes sampin samppr)
```

If desired we can then dichotomize the matrix so that a tie in any of these relations constitutes a tie in the new relation. Otherwise, we could use the raw numbers as an indication of the strength of the positive tie.

Alternatively, we can take an empirical approach and try to discover which relations are highly correlated. In UCINET we can do this by using the

[3] UCINET includes a routine which seeks a 'natural' dichotomization threshold. Essentially, it searches for a cut-off such that the gap between the average value above the cut-off and the average value below the cut-off is as large as possible. This is similar to finding a large drop in a scree plot.

Tools|Similarities procedure to compute Pearson correlations between all pairs of adjacency matrices, and then running Tools|Scaling|Factor Analysis on the correlation matrix to obtain varimax-rotated factor loadings. When we do this with the Sampson data, we find two factors, corresponding to positive and negative ties (see Matrix 5.6). It is worth noting that the positive and negative relations form two separate and orthogonal factors rather than two poles of the same factor. The results suggest it would be sensible to sum the positive matrices and, separately, the negative matrices to obtain two final networks for analysis.

	Factor 1	Factor 2
SAMPLK3	0.864	−0.088
SAMPIN	0.858	−0.088
SAMPES	0.852	−0.115
SAMPLK2	0.844	−0.089
SAMPLK1	0.745	−0.046
SAMPPR	0.739	−0.099
SAMNPR	−0.046	0.624
SAMPDLK	−0.078	0.818
SAMPDES	−0.102	0.889
SAMPNIN	−0.113	0.804

Matrix 5.6 Rotated factor loadings for relations in the Sampson monastery dataset.

5.4.6 Combining nodes

Sometimes we collect data at an individual level, but want to analyze it at a higher level. For example, we collect data on who collaborates with whom in an organization, but what we are really interested in is the pattern of ties between departments (see Figure 5.11). As a result, we want to aggregate the nodes into departments such that a tie between any two nodes becomes a tie between their departments. The inter-departmental ties could be defined as a simple count of the individual-level ties, or we could normalize the count to account for the number of people in each department. One normalization is to divide the count of ties between department A and department B by the number possible, which is simply the size of department A multiplied by the size of department B.[4] These are called densities.

[4] When A and B refer to the same department and the network is non-reflexive, we use $n2 - n$, to account for the impossibility of ties from a node to itself.

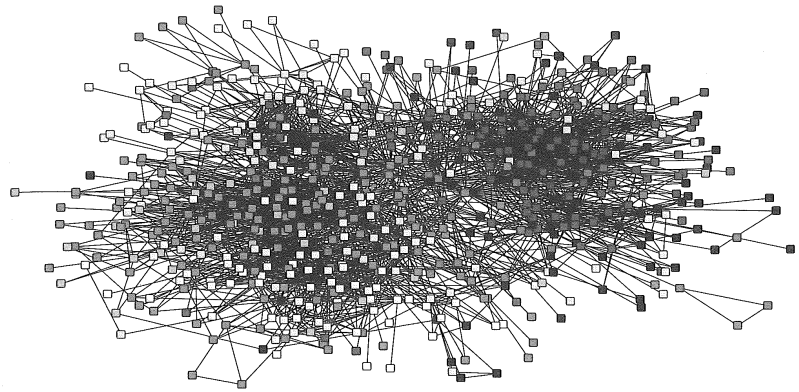


Figure 5.11 Collaboration ties among 960 scientists, shaded by department.

Another normalization is to divide by the total number of ties sent by members of a given department. The resulting matrix gives the proportion of a department's ties that are going to each department (including itself). Still another normalization is to divide the number of ties between departments by the expected value, given a model of independence – that is, nodes make ties without regard for what department they are in. These are the values shown in Matrix 5.7. Regardless of the choice to normalize, the valued matrix can also be dichotomized. The result is a new network – in which the nodes are departments – which can be analyzed in all the usual ways, such as running a subgroup analysis to find out which departments cluster together. Figure 5.12 shows the network of ties based on a dichotomized version of Matrix 5.7.

	BHS	CCG	DCL	ES	HEW	IS	MS	SRG	STAT	TAS
BHS	2.01	1.21	0.89	0.16	0.71	0.20	0.15	3.35	2.62	1.28
CCG	1.21	1.80	0.67	0.30	0.72	0.43	0.40	2.31	2.95	0.80
DCL	0.89	0.67	1.88	0.27	0.63	0.42	0.77	1.87	3.03	0.74
ES	0.16	0.30	0.27	2.79	0.45	1.15	1.80	0.30	1.49	0.30
HEW	0.71	0.72	0.63	0.45	1.47	0.95	0.83	2.03	2.23	0.62
IS	0.20	0.43	0.42	1.15	0.95	3.27	1.32	0.70	2.33	0.24
MS	0.15	0.40	0.77	1.80	0.83	1.32	2.67	0.46	2.01	0.34
SRG	3.35	2.31	1.87	0.30	2.03	0.70	0.46	7.13	4.00	2.53
STAT	2.62	2.95	3.03	1.49	2.23	2.33	2.01	4.00	7.86	2.10
TAS	1.28	0.80	0.74	0.30	0.62	0.24	0.34	2.53	2.10	1.55

Matrix 5.7 Ties between departments. Values are observed counts divided by expected.

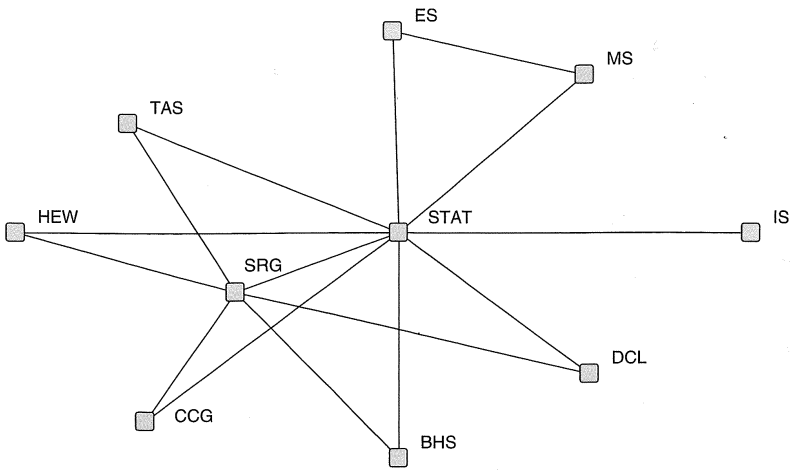


Figure 5.12 Ties between departments based on Matrix 5.7.

5.4.7 Subgraphs

Finally, it may happen that we do not want analyze the whole network. We may wish to delete a node or nodes from the network. This may be because they are outliers in some respect, or because we need to match the data to another dataset where some but not all of the same nodes are present. Or we may wish to combine nodes to form one node that is connected to the same nodes as the individuals were. One reason for combining nodes may be that the data was collected at too fine a level and we need to take a courser-grained analysis. Combining nodes in the same departments would be an example of moving up from the individual level to the department level.

5.5 Normalization

There may be times when we want to re-express, standardize or normalize network data to ensure we are making fair comparisons across rows, columns or entire matrices. In Chapter 4 there was a discussion of the concern that in the use of ratings scales for collecting strength of tie data there could be a problem due to respondents' use and interpretations of the scales. Some respondents may have more readily used the high end of the scale while others used the lower end for reporting on essentially the same tie strength. In another example, if respondents are asked to assess the physical distance from their homes to the homes of all other actors in the network, it may be that some respondents answer

in feet, others in yards, and others in meters. In both of these cases, it is necessary to reduce each row to a common denominator in order to make the data comparable. One way to get around this different scale issues would be to normalize the data. One classic normalization procedure is to compute z-scores for each row so they have a mean of 0 and a standard deviation of 1. There are a range of procedures and one can normalize with respect to means, marginals, standard deviations, means and standard deviations together, Euclidean norms, and maximums. Each type of normalization can be performed on each row separately, on each column separately, on each row *and* each column, and on the matrix as a whole.

5.6 Cognitive social structure data

As introduced earlier in Chapter 2, cognitive social structure data (Krackhardt 1987) is data collected, from each actor in a network, on the perception of who is tied to whom. We can think of this as N adjacency matrices, each $N \times N$ in size. If the relation in question is who likes whom, we can also think of this data as comprising a single three-dimensional matrix LIKE in which LIKE(k, i, j) = 1 if person k perceives that person i has a tie to person j. Note that, depending on how we collected the data, the perceived ties may be directed so that LIKE(k, i, j) need not equal LIKE(k, j, i).

Typically, one of the reasons for collecting data of this type is to investigate the accuracy of people's perception of the network. For example, Krackhardt (1987) found that managers who had more accurate views of the network had more power in the organization. However, measuring accuracy entails defining a right answer – the 'true' network. There are several approaches to doing this. Perhaps the most commonly used is what Krackhardt calls the 'row-dominated locally aggregated structure' (RLAS). In the RLAS, we assume that the person sending a tie (such as trust) to another is the authority on that tie, and we take their word for it. This means that to construct the RLAS matrix, we simply define RLAS(i, j) to be equal to LIKE(i, i, j). The RLAS is the matrix you would have obtained had you simply collected ordinary network data.

Another way to construct the true matrix is the intersection method. Here the idea is that a tie exists from i to j if i says they have a tie to j, and j perceives an incoming tie from i. This is not reciprocity, which would be that i says they like j and j says they like i. Rather, it is agreement: i says they like j, and j says that i likes j. This matrix can be defined as ILAS(i, j) = 1 if LIKE(i, i, j) = 1 and LIKE(j, i, j) = 1. The logic of the intersection approach is that you have more confidence in a tie if both people involved report that tie.

An alternative approach is a consensus method that takes into account the perceptions not just of the two people involved in a tie, but everyone else's as well. For example, we could use a threshold model that declares a tie from i to j if at

least a certain percentage of people (say, 50%) see it that way. There are a number of more sophisticated versions of this approach as well (Romney, Weller and Batchelder 1986; Butts 2003).

Once a 'true' network is constructed, we can then evaluate the similarity of each person's individual perceived network with the true network. This is simply a matter of measuring the similarity between each person's perceived matrix and the adjacency matrix of the true network.

5.7 Matching attributes and networks

A quirk of network analysis programs such as Pajek and UCINET (but not NetDraw) is that they identify nodes by position in the data file rather than using unique identifiers. They allow node labels, such as 'John Smith' and '101–2', but the way they identify nodes internally is simply by their ordinal position in the list of nodes. As a result, if you delete node 15, what used to be node 16 will now be seen as node 15 in the program.

Among other things, this means that to combine network and attribute data in a single analysis, you must be careful to ensure that the nodes are in the same order in the two files. For example, if the data consists of Matrices 5.8 (a network) and 5.9 (an attribute matrix), an analysis that uses the network and the attribute data together (incorrectly) assumes that the person in the second row of the network matrix (Jeff) is female, because the second person in the attribute matrix is female. Similarly, if you have attribute data on 1000 people but collected network ties only among a subset of 50 people, a network analysis program like UCINET (unlike NetDraw) will not know how to look up the values of the 50 people in the larger attribute dataset.

		1	2	3	4	5	6	7	8	9	10
		S	J	M	V	R	C	A	A	J	F
1	Steve	0	0	1	0	0	0	0	0	0	1
2	Jeff	1	0	1	0	0	0	0	0	0	0
3	Martin	0	1	0	0	0	0	1	1	0	0
4	Vanessa	0	0	0	0	0	1	0	0	0	0
5	Roberta	0	0	0	0	0	0	0	0	1	1
6	Chris	0	0	0	0	0	0	0	0	0	0
7	Ann	0	0	0	0	0	0	0	0	0	0
8	Adam	1	1	1	1	0	1	0	1	0	0
9	James	1	1	0	0	0	0	1	0	0	1
10	Fiona	0	0	0	0	0	0	0	0	0	0

Matrix 5.8 A directed network.

		Age	Gender	Income
1	Adam	19	1	489
2	Ann	22	2	121
3	Chris	61	2	239
4	Fiona	37	2	3125
5	James	20	1	1421
6	Jeff	16	1	1030
7	Martin	44	1	1453
8	Roberta	26	2	3412
9	Steve	43	1	2050
10	Vanessa	35	2	245

Matrix 5.9 Attribute matrix of Matrix 5.8 in alphabetical order.

As a result, the researcher needs to take steps to ensure their attribute datasets are properly matched to their network datasets, and of course, if one has different matrices for different social relations, such as a friend network and an advice network, these need to be matched as well. An easy way to do this in UCINET is to run 'Match Net and Attrib datasets' or 'Match Multiple Datasets', as shown in Figure 5.13. This generates new versions of all the input datasets, whether attribute or network, that are matched by node labels. Extra nodes found in some files but not others are ignored.

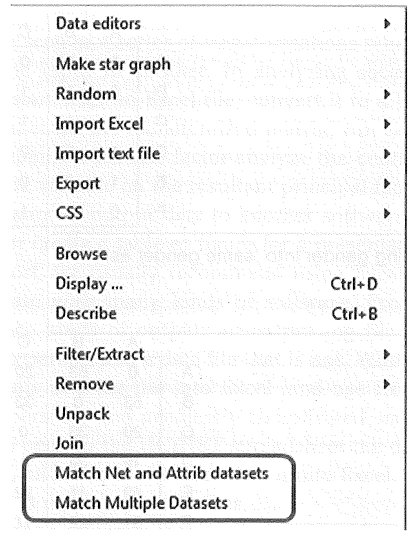


Figure 5.13 Data menu in UCINET.

5.8 Converting attributes to matrices

In network analysis we frequently need to make a matrix out of an attribute. In general, what we are doing is changing a node quality into a relational quality, so that instead of height we have 'is taller than' and instead of gender we have 'is same gender as'. Often the purpose is to use the 'matrified' attribute as a dyadic predictor of ties in a QAP regression (Chapter 8). For example, suppose we have a directed relation such as 'likes' and we want to predict who likes whom. One idea is that people like people who are similar to themselves on socially significant attributes such as race, gender, age, and status. For race and gender we can construct 'same race as' and 'same gender as' matrices, as shown in Figure 5.14. For age, we might use absolute difference in age, as shown in Figure 5.15. For status, we might anticipate that low-status individuals have a bias for higher-status individuals and against lower-status individuals. In this case we might use simple difference in status, as shown in Figure 5.16. Alternatively, we might argue that the probability of liking someone increases with their status, even if that status is still lower than one's own. In that case what we want to do is to create a matrix in which the values in each column equal the status of the node corresponding to that column. Effectively, it is like writing the status vector as a row vector and then copying it $n - 1$ times. This is shown in Figure 5.17.

Actor	Gender
A	1
B	2
C	2
D	1
E	1
F	2

	A	B	C	D	E	F
A	1	0	0	1	1	0
B	0	1	1	0	0	1
C	0	1	1	0	0	1
D	1	0	0	1	1	0
E	1	0	0	1	1	0
F	0	1	1	0	0	1

Figure 5.14 Converting gender into 'same gender as'.

Actor	Age
A	14
B	67
C	34
D	33
E	56
F	45

	A	B	C	D	E	F
A	0	53	20	19	42	31
B	53	0	33	34	11	22
C	20	33	0	1	22	11
D	19	34	1	0	23	12
E	42	11	22	23	0	11
F	31	22	11	12	11	0

Figure 5.15 Converting age into 'difference in age'.

Actor	Status
A	6
B	10
C	3
D	5
E	9
F	4

	A	B	C	D	E	F
A	0	−4	3	1	−3	2
B	4	0	7	5	1	6
C	−3	−7	0	−2	−6	−1
D	−1	−5	2	0	−4	1
E	3	−1	6	4	0	5
F	−2	−6	1	−1	−5	0

Figure 5.16 Converting status into relative status using a simple difference.

Actor	Status
A	6
B	10
C	3
D	5
E	9
F	4

	A	B	C	D	E	F
A	6	10	3	5	9	4
B	6	10	3	5	9	4
C	6	10	3	5	9	4
D	6	10	3	5	9	4
E	6	10	3	5	9	4
F	6	10	3	5	9	4

Figure 5.17 Converting status into 'status of alter'.

5.9 Data export

Data analysis can be seen as a series of transformations where the output of one analysis becomes the input to another. In analyzing social network data, for example, we might start with an Excel file, convert it to a UCINET system file, dichotomize it, symmetrize the dichotomized matrix, run centrality on the symmetrized and dichotomized matrix, factor-analyze the centrality measures, and regress some outcome variable on the resultant principal factor. Anywhere along the line, we might want to output data to another software package (such as a statistical package) or create a table or figure for a presentation or paper.

As with data import, we usually recommend using Excel as an intermediary that can communicate with many kinds of software. Programs like UCINET typically produce two kinds of output: an output log file which is a text file meant for humans to peruse, and a data file that is not. While it is possible to cut and paste the contents of a log file into Excel (and use Excel's text-to-columns feature), it is a cumbersome and needlessly complicated way to transfer data. A better approach is to open in the UCINET spreadsheet the data file that was created by any analysis, and then to cut and paste it into Excel. This spreadsheet-to-spreadsheet transfer is much more efficient.

Sometimes we want to reformat the data before transferring it to another program. For example, suppose we are analyzing longitudinal network data, such as

friendships among 17 college men measured at 15 points in time (this the Newfrat dataset in UCINET). The data is stored in a single 'stacked' UCINET dataset consisting of 15 person-by-person matrices that can be thought of as layers or slices in a three-dimensional data object. If we run, say, centrality on a dichotomized version of this dataset, the program will produce a new stacked dataset containing the centrality scores for each person on each centrality measure for each time point, arranged as 15 person-by-measure matrices. To analyze this using, say, a random effects regression in Stata, we execute the following series of steps.

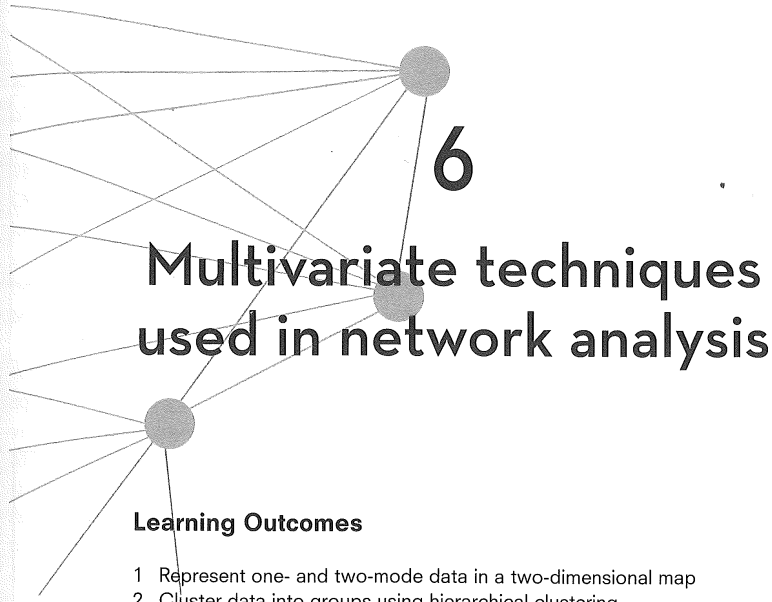
First, join all the matrices together 'vertically' to create a new matrix that has 255 rows and as many columns as there centrality measures (say, six columns). The rows are person–time pairs which are appropriate for longitudinal analyses. We can do this in UCINET's Matrix Algebra command-line environment by typing the following:

```
-->newmat = appendasrows('newmat-cent')
```

The result is a new matrix called newmat which can be opened in the UCINET spreadsheet, and pasted into Excel, and then transferred to Stata. Before transferring to Stata, however, it is useful to open another matrix – automatically created by APPENDASROWS and named newmat-id – in the UCINET spreadsheet, and cut and paste the contents into the Excel file. The newmat-id matrix consists of two variables that give the person-ID and the time-period-ID for each of the 255 cases.

5.10 Summary

Network data needs to be in specific formats in order to efficiently enter it into computer programs. Most networks are sparse, so list formats are one of the most efficient ways to enter data. Binary data can be formatted as nodelists, whereas valued data requires edgelists. We normally have attribute information on the nodes of a network. This information is in the form of full matrices or vectors and can be imported using spreadsheet formats. Once imported, data should be examined and cleaned, if necessary, to make sure it is as accurate as possible. Then various transformations such as symmetrizing, dichotomizing, normalizing and transposing can be applied. Note that if the transformations have fundamentally changed the nature of the relation, this must be taken into account when interpreting the output obtained from any analysis. For example, if an advice network has been symmetrized via the maximum method, we can no longer interpret x_{ij} as indicating that node i gives advice to node j. However, we can reasonably regard x_{ij} as indicating that i and j are involved in an exchange of information.



6 Multivariate techniques used in network analysis

Learning Outcomes

1 Represent one- and two-mode data in a two-dimensional map
2 Cluster data into groups using hierarchical clustering
3 Correctly interpret the information contained in the clusters and maps

6.1 Introduction

In this chapter we briefly introduce the reader to a number of data analysis techniques that are not specific to network analysis, but are often used as an integral part of network analysis procedures. We use these throughout the book. For the most part, they consist of exploratory multivariate statistics, such as multidimensional scaling, correspondence analysis and hierarchical clustering.

6.2 Multidimensional scaling

The purpose of multidimensional scaling (MDS) is to provide a visual representation of the pattern of proximities among a set of objects. By proximities we mean any symmetric, one-mode matrix of similarities, tie strengths, dissimilarities, distances, etc. among a set of objects. MDS places points (corresponding to our objects) in space such that the distances between the points correspond in a predetermined way to the proximities among objects in the data. For example, if the input data is physical distances between US cities, what MDS will do is to draw a map of the USA. Matrix 6.1 gives an example of driving distances between nine US cities. An MDS map based on that data is shown in Figure 6.1.