

Appendix 1

Getting Started with Pajek

A1.1 Installation

Pajek software for network analysis can be installed on all computers operating under Windows 95, 98, 2000, NT, or XP. The software can be installed on the computer's hard disk: double-click the file named `Pajek.be.exe` (Pajek book edition) and follow the instructions.

Pajek can be started by double-clicking the file `Pajek.exe`, which you will find in the directory where you installed the program. The subdirectory *Doc* contains additional information about the software in several hypertext files (`.htm`), which can be opened in an Internet browser, and Portable Document Files (`.pdf`), which can be read with (Adobe) Acrobat Reader.

A1.2 Network Data Formats

*File>Network
>Read*

Pajek can read network data in several plain text formats, that is, files containing unformatted text (ASCII). We briefly discuss them in the order in which they appear on the dialog box issued by the *File>Network>Read* command (Figure 128).

The first two data formats are indigenous Pajek formats: Pajek networks and Pajek matrices. The *Pajek network* format (the filename extension is `.net`) is explained in Chapter 1 (Section 1.3.1) and additional features for longitudinal networks are discussed in Chapter 4 (Section 4.5). Basically, it is a list of vertices followed by a list of arcs and edges. It is the most flexible format because it allows for multiple lines and many (layout) properties of the vertices and lines can be specified, which are explained in the file `DrawEPS.htm` on the book's Web site. The Pajek network format is in line with the logic of relational databases. See Section A1.3.3 for more information on organizing your network data as a relational database and exporting from database software to Pajek. There are three network formats: Arcs/Edges, ArcsList/EdgesList, and ArcsList/EdgesList (min). In Section A1.3, we discuss only the first type (Arcs/Edges) because it is most flexible.

The *Pajek matrix* format (with the filename extension `.mat`) is slightly different. The list of vertices is the same as in the Pajek network format

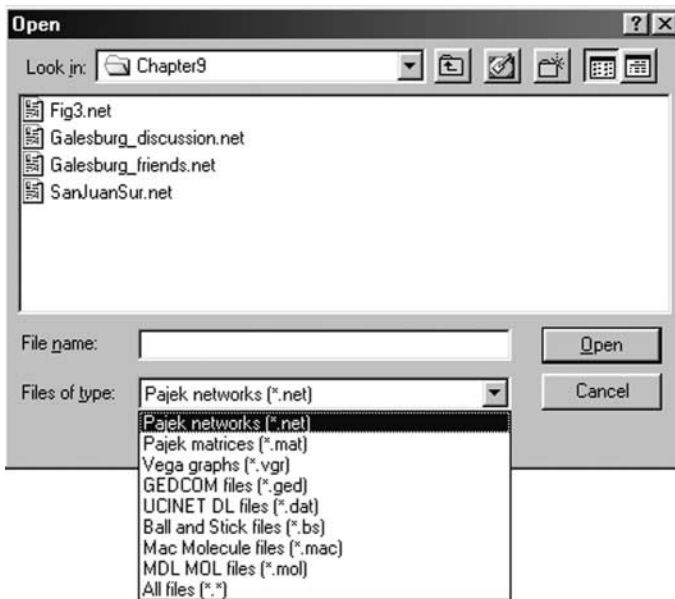


Figure 128. Read Network dialog box.

but the list of edges and arcs is replaced by a matrix consisting of integers or real numbers, which are separated by blanks (Figure 129). This is an ordinary adjacency matrix (see Chapter 12, Section 12.2). The Pajek matrix format is useful for importing network data in matrix format.

The third data format is the *Vega* format devised by Pisanski. The fourth format, *GEDCOM*, is the standard data format for genealogical data, which is discussed in Chapter 11 (Section 11.3, see also Further Reading). The fourth format, *UCINET DL files*, is the raw data format of another widely used program for network analysis: UCINET. Both UCINET and Pajek can read networks exported in this format, so this is the format for exchanging data between the two software packages. For more information about UCINET, consult the Web site <http://www.analytictech.com/>.

The last three data formats (*Ball and Stick*, *Mac Molecule*, and *MDL MOL*) were developed for chemistry. These formats are not widely used in social network analysis.

```
*Vertices      6
 1 "Ada"          0.1646   0.2144   0.5000
 2 "Cora"          0.0481   0.3869   0.5000
 3 "Louise"        0.3472   0.1913   0.5000
 4 "Jean"          0.1063   0.5935   0.5000
 5 "Helen"         0.2892   0.6688   0.5000
 6 "Martha"        0.4630   0.5179   0.5000

*Matrix
0.000 1.000 1.000 0.000 0.000 0.000
1.000 0.000 0.000 1.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000
0.000 0.000 0.000 0.000 1.000 0.000
0.000 0.000 0.000 1.000 0.000 0.000
0.000 0.000 0.000 0.000 0.000 0.000
```

Figure 129. A network in Pajek matrix format.

A1.3 Creating Network Files for Pajek

There are several ways to create data files that can be read by Pajek. We distinguish between three methods: creating data files manually within Pajek, creating them in a word processor, and exporting them from a relational database. We present the methods in this order, from the simple and inflexible to the complicated and versatile.

A1.3.1 Within Pajek

*Net>Random
Network>Total
No. of Arcs*

In Pajek, you can create new networks, partitions, and vectors manually by means of a number of commands and tricks, which have been presented in the chapters of this book. Let us start with creating a new network. We suppose that you have the data on the vertices and lines at hand (e.g., on paper). Start Pajek and use the *Net>Random Network>Total No. of Arcs* command to create a new network (see Chapter 1, Section 1.4). Enter the desired number of vertices in the first dialog box and request zero arcs in the second dialog box because you are going to add the lines manually.

*Partition>Create
Null Partition*

*File>Partition>
Edit*

As a first step, add labels to the vertices so they can be identified more easily. Here, the trick is to create a new partition with the *Partition>Create Null Partition* command (see Chapter 2, Section 2.3) and then open it in an Edit screen (the *File>Partition>Edit* command, see Chapter 2, Section 2.3). The result may look like that in Figure 130. In this screen, you can manually edit the labels of the vertices. Now you have defined the labels of the vertices.

*File>Network>
Edit*

*Editing Network
screen*

As a next step, you must add lines to the network. This can be done in the Editing Network screen (Figure 131), which can be opened either from the Main screen with the *File>Network>Edit* command specifying a vertex number or label or from the Draw screen by right-clicking a vertex. Double-click the word *Newline* to add an edge (enter the number of the vertex that must be connected to the selected vertex) or an arc [enter the vertex number preceded by a plus sign (arc toward the selected vertex) or a minus sign (arc from the selected vertex)]. This is discussed in more detail in Chapter 1, Section 1.4.

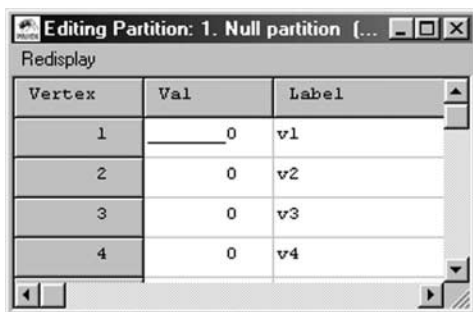


Figure 130. Editing vertex labels.

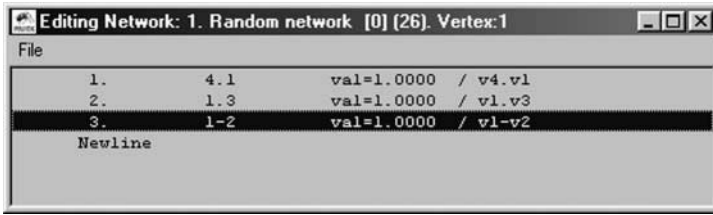


Figure 131. Edit Network screen.

If you have a discrete characteristic of the vertices (e.g., the sex of persons), you can store this in a new partition. Create a new empty partition and open it in an Edit screen as discussed above. Then give each person (vertex) the appropriate sex code in the Edit Partition screen. Just enter the right code in the “Val” column (see Figure 130 and Chapter 2, Section 2.3). Note that Pajek accepts only numerical codes. As an alternative, you can draw the network with a new empty partition by executing the *Draw>Draw-SelectAll* command (see Chapter 2, Section 2.3). Now, you can respectively raise and lower the class associated with a vertex by left-clicking the vertex while pressing the *Shift* key, which is equivalent to clicking with the middle mouse button, or left-clicking it while holding down the *Alt* key (see Chapter 2, Section 2.3).

Vectors, which specify continuous characteristics of vertices, may be created in a similar way. Create a new vector with the *Vector>Create Identity Vector* command and edit it manually in the Editing Vector screen, which opens on execution of the *File>Vector>Edit* command. Initially, all vertices have one as their vector value. You can manually change it to the desired number.

Don’t forget to save the data files before you exit from Pajek, otherwise your work will be lost. Pajek does not save data files automatically. Save the data files by means of the diskette button to the left of the drop-down menus or with the *Save* command in the appropriate submenu of the *File* menu: *Network*, *Partition*, or *Vector*. You can save the network in any of Pajek’s formats as well as in UCINET’s DL format.

File>Partition>Edit

Draw>Draw-SelectAll

Change the class number of vertices

Vector>Create Identity Vector

File>Vector>Edit

File>Network>Save

File>Partition>Save

File>Vector>Save

A1.3.2 Word Processor

It is possible to create a network data file in Pajek, but adding lines by means of the Editing Network screen (Figure 131) is quite tedious. Usually, it is more efficient to create a network with the right number of vertices but without lines in Pajek with the *Net>Random Network>Total No. of Arcs* command and to change vertex labels and add lines in a word processor. Note: never use tabs!

Figure 132 shows an empty network consisting of six vertices, which was created with the *Net>Random Network>Total No. of Arcs* command and saved from Pajek as a Pajek Arcs/Edges network file. It is a plain text file so it can be opened in any word processor (e.g., NotePad and WordPad). Its first line specifies the number of vertices. This number may never be lower than the highest vertex number in the network.

Net>Random Network>Total No. of Arcs

```

*Vertices      6
  1 "v1"      0.1000  0.5000  0.5000
  2 "v2"      0.3000  0.1536  0.5000
  3 "v3"      0.7000  0.1536  0.5000
  4 "v4"      0.9000  0.5000  0.5000
  5 "v5"      0.7000  0.8464  0.5000
  6 "v6"      0.3000  0.8464  0.5000

*Arcs

```

Figure 132. An empty network in Pajek Arcs/Edges format.

The first line is followed by the list of vertices: each vertex has one line containing the vertex number (here: 1 to 6), the vertex label, which is the letter V followed by a number by default, and the x , y , and z coordinates of the vertex in a layout. In a word processor, you can easily change these properties of the vertices but it is wise not to meddle with the vertex numbers even though Pajek allows vertices to be out of order or some vertex numbers to be absent in the list. Empty lines, however, are not allowed in the list of vertices.

In Figure 132, the last line reads **Arcs*, which signals the start of the list of arcs. In this example, the list is still empty. You may add arcs manually, using one line for each arc. First, specify the number of the sending vertex. Next, add one or more spaces and the number of the receiving vertex. Optionally, you can add the line value after one or more spaces. If you do not specify the line value, it is considered to be 1 (e.g., the arc from vertex one to vertex two in Figure 133). For undirected lines, first add the line **Edges* and then use a line for each edge just like the arcs. The only difference is that it does not matter which vertex is specified first. There is no limit to the number of arcs or edges, and multiple arcs and edges are allowed. The order of the lines does not matter.

If you prefer to enter the data as a matrix, as in Figure 129, replace the line **Arcs* with **Matrix* and type in a square matrix, that is, a matrix with as many rows as columns. Use one line for each row and at least one space between the cells in different columns. You may use any number of spaces between two numbers within a row. Now, the network data file may look like that in Figure 134: there is an arc from vertex 1 to vertex 2 (second number or cell in the first row of the matrix) with a line value of 1. Line values may have decimal places and negative signs.

```

*Vertices      6
  1 "v1"      0.1000  0.5000  0.5000
  2 "v2"      0.3000  0.1536  0.5000
  3 "v3"      0.7000  0.1536  0.5000
  4 "v4"      0.9000  0.5000  0.5000
  5 "v5"      0.7000  0.8464  0.5000
  6 "v6"      0.3000  0.8464  0.5000

*Arcs
  1 2
  2 5 2.5
  4 5
*Edges
  3 4 -1
  5 6

```

Figure 133. A network in the Pajek Arcs/Edges format.

```

*Vertices      6
  1 "v1"      0.1000  0.5000  0.5000
  2 "v2"      0.3000  0.1536  0.5000
  3 "v3"      0.7000  0.1536  0.5000
  4 "v4"      0.9000  0.5000  0.5000
  5 "v5"      0.7000  0.8464  0.5000
  6 "v6"      0.3000  0.8464  0.5000

*Matrix
0  1  0  0  0  0
0  0  0  0  2.5 0
0  0  0 -1  0  0
0  0 -1  0  1  0
0  0  0  0  0  1
0  0  0  0  1  0

```

Figure 134. A network in the Pajek matrix format.

There are some disadvantages to the matrix data format. First of all, Pajek cannot distinguish between an edge and a bidirectional arc. Therefore, a matrix read by Pajek only contains arcs. Second, Pajek cannot distinguish between a line with a value of zero and an absent line, so Pajek replaces all zeros with arcs with a line value of zero unless it is instructed to ignore lines with a value of zero in the *Threshold* field of the *Options>Read/Write* dialog box. The threshold must be set to zero before a network is opened. Note that the threshold applies to absolute values: negative line values (e.g., -1 in Figure 134) are not eliminated if the threshold is zero. Finally, it is impossible to have multiple lines in matrix format.

[Main]
Options>Read/
Write>Threshold

Until now, we have considered only one-mode networks. Section 5.3 in Chapter 5, however, introduced *two-mode networks*: networks consisting of two sets of vertices such that all lines are found between the sets (e.g., affiliations of people to organizations). Pajek network data files have a special arrangement for two-mode networks. You can split the list of vertices into two sets or modes in the **Vertices* statement by specifying the total number of vertices followed by one or more spaces and the number of vertices in the first mode. This requires that the list of vertices is sorted such that the vertices in the first mode have the lowest vertex numbers and the vertices in the second mode have the highest numbers. Figure 135 shows the data file of a two-mode network consisting of two organizations and four persons. Note that Pajek will issue a warning if it encounters lines within a mode, which should not occur in a two-mode network.

```

*Vertices      6 2
  1 "org1"      0.1000  0.5000  0.5000
  2 "org2"      0.3000  0.1536  0.5000
  3 "person1"   0.7000  0.1536  0.5000
  4 "person2"   0.9000  0.5000  0.5000
  5 "person3"   0.7000  0.8464  0.5000
  6 "person4"   0.3000  0.8464  0.5000

*Arcs
1 3
2 5 2.5
1 4
*Edges
2 4 -1
1 6

```

Figure 135. A two-mode network in the Pajek Arcs/Edges format.

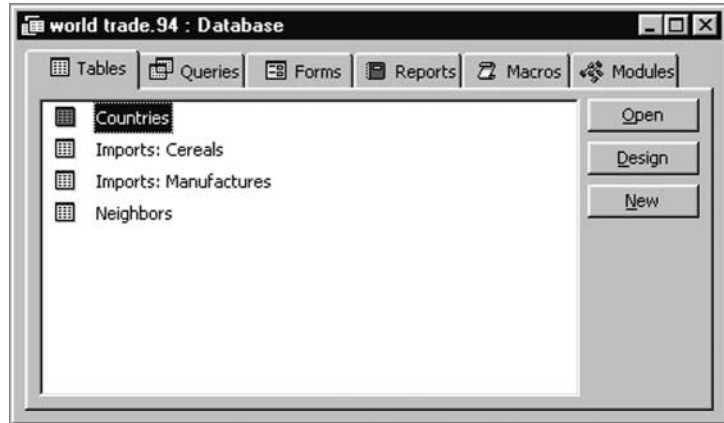


Figure 136. Four tables in the world trade database (MS Access 97).

A1.3.3 Relational Database

Data on large networks are optimally stored as relational databases. In a relational database, two tables can represent a *one-mode network* (Figure 136): one table contains the vertices of the network (e.g., countries) and another table stores the lines between the vertices (e.g., trade relations such as imports). This matches the basic structure of the Pajek Arcs/Edges network format: a list of vertices and a list of lines.

This book's Web site contains a Microsoft Access database with the world trade data of Chapter 2 (see Section 2.10 for the data sources): `world-trade-94.mdb`. The database is made with MS Access version 97, so you have to convert the database if you open it in a newer version of this software. The *Countries* table is the heart of the database, containing one record (row) for each country (Figure 137). Each country has a numerical code, which is its vertex number in the network. In addition to the country's name, several properties are listed, including its GDP, geographic location, and population growth. The table provides the data for the list of vertices in a network data file and the data for partitions

vertexnr	vertexlabel	x	y	continent	trade bloc	position
1	Algeria	0.4380682	0.4957265	1	0	999998
2	Argentina	0.2159091	0.7749288	6	3	2
3	Australia	0.8636364	0.7008547	5	0	2
4	Austria	0.46875	0.411396	3	0	2
5	Barbados	0.2670454	0.5766382	6	4	999998
6	Bangladesh	0.7301137	0.5071225	2	8	999998
7	Belgium, Lux.	0.45	0.3931624	3	1	1
8	Belize	0.1352273	0.5259259	4	4	999998
9	Bolivia	0.2215909	0.6951567	6	3	999998
10	Brazil	0.2897727	0.6780627	6	3	2
11	Canada	0.1363636	0.3304843	4	2	1
12	Chile	0.1931818	0.7635328	6	3	3
13	China	0.75	0.4643874	2	0	999998

Figure 137. Contents of the *Countries* table (partial).

tail	head	value	valperc
Belgium, Lux.	Southern Africa	9931	2.136122
France	Southern Africa	12892	2.773022
Germany	Spain	29006	6.239084
United Kingdom	Sri Lanka	20081	4.319349
Australia	Sweden	7514	1.616234
Canada	Switzerland	25318	22.98711
United States	Thailand	70774	64.25822
Chile	Trinidad Tobago	1902	1.726893
Colombia	Tunisia	4836	4.390775
Germany	Ecuador	5062	4.595969

Figure 138. A Lookup to the *Countries* table.

and vectors (e.g., GDP and population size of the countries). In principle, any relevant characteristic of the countries can be added to this table. For instance, we added the coordinates of the country in a map (fields x and y), which we used to create the geographical layout of the trade network in Figure 10 (see Chapter 2).

The network lines are stored in separate tables. We collected the information on two kinds of trade: imports of miscellaneous manufactures of metal (table *Imports: Manufactures*) and imports of cereals (table *Imports: Cereals*). Both tables contain one record for each trade tie with the code of the exporting country (field *tail*) and the code of the importing country (field *head*). The value of imports (in 1,000 U.S.\$) and the share in a country's total imports of this commodity have been registered. These tables are very similar to the arcs list in a Pajek network file. In addition, we created a table of undirected ties (viz., countries that share a border (table *Neighbors*). This table contains a record for each pair of neighboring countries and it is the source of a list of edges in a network data file.

The *Imports: Cereals* table exemplifies a feature of relational database software that is very useful for efficient data entry. Once the table of vertices has been created and filled with data, references to this table can be made from another table. In a table containing lines between vertices, you can look up the (vertex) number of a country directly. In MS Access, for example, the countries can be displayed as a list, from which you can easily pick the right country: start typing the country's name and the software will lead you to the country you are looking for (Figure 138). When you pick a country from this list – press *Enter* when the country is found – the country's (vertex) number is added to the table of ties although its name is displayed. This helps avoiding data entry errors.

Once all data have been entered into the database, Pajek network data files can be created. This boils down to combining the table of the vertices (*Countries*) with one or more tables of ties (e.g., *Imports: Manufactures* and *Neighbors*) in the right layout. In Microsoft Access, the Report utility is most suited for that. If you open the list of reports in the example database, you will see several reports (Figure 139). For exporting a one-mode network, two reports are needed. The first report (*List of vertices*)



Figure 139. Export a report to plain text.

lists the vertices in the right layout and adds the **Vertices* line. The second report (*List of arcs*) contains the formatted list of arcs with the **Arcs* line. In the case of an undirected relation, a report is needed starting with **Edges* instead of **Arcs* (report *List of edges*). The two reports (or three reports in case there are arcs and edges simultaneously) are combined in a master report (*Export network to Pajek: just arcs*, *Export network to Pajek: just edges*, and *Export network to Pajek: arcs and edges*), which merely concatenates the partial reports.

The master report must be exported to plain text: text without special layout characters. Unfortunately, Microsoft Access is a little uncooperative at this point. It is possible to export the report in plain text format. Right-click the report in the report list (Figure 139) and select *Save As/Export* and subsequently *To an External File or Database* and finally save it with *Text Files* selected as the *Save as type*. Sometimes, however, the saved file has an empty line every four lines, which renders it unreadable to Pajek.

As an alternative, the report can be exported to Microsoft Word (in Rich Text format) with the *Tools>Office Links>Publish It with MS Word* command. Now Access adds page breaks, which must be deleted before the file is saved in plain text format. The Rich Text format file is automatically opened in MS Word, where you can delete the page breaks with the command *Replace* (or *Ctrl h*). Enter *^m*, which denotes a page break, in the *Find what:* line and replace it with nothing (leave the *Replace with:* line empty). Finally, use the command *File>Save As* and select the *Plain text (*.txt)* type to save the network data file. Give it the *.net* extension so Pajek can recognize it as a network data file.

Exporting partitions and vectors is less complicated but more sensitive to errors. Partitions and vectors are simply sorted lists of numbers, one number for each vertex on a separate line, preceded by a **Vertices*

statement. A simple report listing the sorted contents of a field in the table of vertices is all you need to create a partition or vector data file; the reports *Export a partition* and *Export a vector* do this. They can be exported to plain text files in the manner just described. Use the `.clu` and `.vec` extensions to enhance recognition by Pajek.

However, the user should be careful. The report can export nonnumeric fields but Pajek cannot handle these. This will be apparent when you try to open a nonnumeric partition or vector in Pajek. There is an even more important limitation. The report lists just the values sorted by vertex number. The value attached to the lowest vertex number is stored in the first line after the `*Vertices` statement. Reading the partition or vector, Pajek will assume that this value belongs to vertex 1. Therefore, vertex numbers in the database must start with 1 for a partition or vector to match the network. For the same reason, no vertex numbers may be missing in the exported network, so vertex numbers must range from 1 to the number of vertices.

This is required for a partition or vector exported from the database to match a network exported from the same database. Otherwise, the partition or vector will contain fewer vertices than the network, so you cannot combine them in Pajek. If you want to export only some of the vertices in the database, you should either renumber the remaining vertices in the database (in the vertices table as well as in the relations tables!) or export the entire network, partitions, and vectors and remove the unwanted vertices from the network, partitions, and vectors in Pajek (see Chapter 2).

As a final note on exporting vectors (and coordinates in the list of vertices), we want to draw your attention to the language and regional settings of Windows. If decimal places are not separated by a point, Pajek will not be able to read the numbers. You must either adjust your regional settings or replace your local decimal separator by a point by means of a word processor in your data files.

Exporting a *two-mode network* such as corporate interlocks is more complicated. In a relational database, one would probably store the vertices in two tables: one containing the first mode, for instance, the companies (table *Company* in Figure 140), and the second mode, for instance,

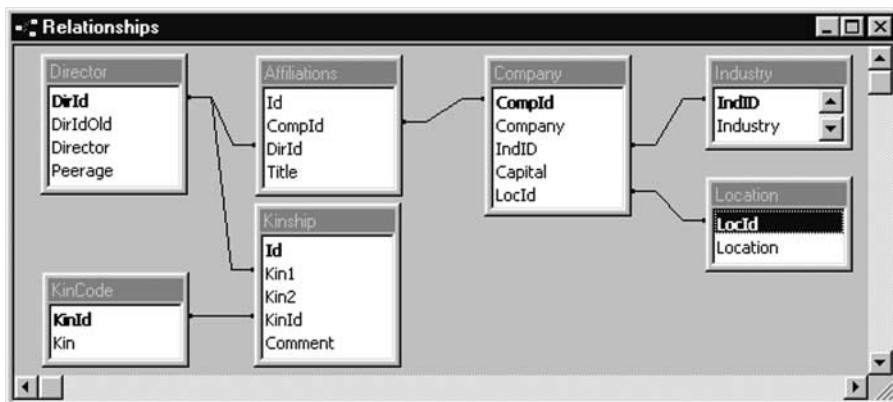


Figure 140. Tables and relations in the database of Scottish companies.

the directors, in another table (table *Director* in Figure 140). Ties connect vertices from one table of vertices to the other (table *Affiliations* in Figure 140). In Pajek, the vertex numbers of the two modes should not overlap. One should number the vertices in the first mode (the firms) from 1 to the number of vertices. The numbering of the other mode (the directors) should start with 1 plus the number of vertices in the first mode. If this precaution is taken (in the example database, queries automatically perform this task), a two-mode network can be exported to Pajek along the lines sketched above, with the additional trick of merging the two tables of vertices into one list.

The example database *Scotland.mdb* contains the data from John Scott's study of interlocking directorates (the example of Chapter 5) and a report for exporting a two-mode network (*Export two-mode network to Pajek*). In this example, special queries ensure that the vertex numbers of directors and companies do not overlap.

In the Pajek Web site, we included an empty MS Access (97) database (*network.mdb*) that you can use as a starting point for your own data collection. The database contains the tables, queries, and reports you need to create Pajek data files for one-mode and two-mode networks. The tables contain a minimum of sample data, which can easily be replaced by your own data. Note that vertex numbers are automatically assigned to vertices in the tables. Do not delete a vertex and its number because that number cannot be used afterwards. Furthermore, do not change the names of the reports, tables, and fields within the tables unless you know very well what you are doing. Renaming may affect the reports and the links between tables. To make the most of the relational database, learn more about the database software or find someone who can advise you.

A1.4 Limitations

Pajek can handle networks up to 9,999,997 vertices. In general, drawings of networks should contain no more than a few thousand vertices because for very large networks the drawing procedure is time consuming and the drawing is visually unattractive. By default, Pajek will not draw networks larger than 5,000 vertices but you can change this limit in the *Options>Read/Write* menu. Available resources on the computer may pose additional limitations.

A1.5 Updates of Pajek

Updates of Pajek can be downloaded from <http://vlado.fmf.uni-lj.si/pub/networks/pajek/default.htm>. Due to modifications, newer versions of Pajek may not match the illustrations, command names, and output described in this book exactly.

Appendix 2

Exporting Visualizations

In Chapter 1, several options for exporting graphical output were briefly discussed. We provide more details in this appendix. In Section A2.1, we discuss the graphical formats that Pajek exports to. Next we discuss the options to adjust the layout of the exported image (Section A2.2). *[Draw] Export*

A2.1 Export Formats

Pajek can save a sociogram in six different graphical formats. In most cases, viewers and plug-ins must be installed on your computer before you can display the exported layout. We now discuss each graphic format, how it is exported from Pajek, and how it can be viewed and edited. Note that all references to software and Web sites are made to the year 2004; software updates may have changed and Web sites may have disappeared since then.

A2.1.1 Bitmap

A bitmap exported from Pajek with the *Export>Bitmap* command is merely a screen shot of the Draw screen in its current state, which is saved to a file. Because the bitmap is a very general graphical format, it can be viewed and embedded in most Windows software. A bitmap can be edited in any Windows paint program but editing is cumbersome because you have to edit each pixel (screen point) separately. It is not possible to move entire vertices, lines, or labels. The bitmap format is useful mainly for a quick presentation of a sociogram. For quality images, use Encapsulated PostScript or Scalable Vector Graphics. *Export>Bitmap*

A2.1.2 Encapsulated PostScript

Encapsulated PostScript (and ordinary PostScript) produces two-dimensional vector drawings: each vertex, line, and label is defined as a separate shape with a particular location in the plane. These formats have been developed for quality printing. The drawing or any part of it can be enlarged or reduced without loss of quality. All figures in this book, except for the screenshots, are based on Encapsulated PostScript from

Export>EPS/PS

Pajek. Note that partition cluster numbers, which are displayed within vertex labels in the Draw screen, are automatically replaced by numbers inside the vertices in PostScript output (e.g., see Figure 31 in Chapter 3).

If you export a network layout in Encapsulated PostScript or PostScript formats (the *Export>EPS/PS* command), you can choose among several file types in the *Save As* dialog box. On the one hand, you can choose between Encapsulated PostScript (EPS) and ordinary PostScript (PS). We advise using Encapsulated PostScript always; ordinary PostScript cannot be directly viewed or printed.

On the other hand, you can choose between *What You See Is What You Get* (WYSIWYG), *Clip*, and *NoClip*. This choice affects the size and boundary of the exported drawing. In the WYSIWYG export, a vertex or label located outside the boundaries of the Draw screen will not be visible in the drawing (although it is defined in the drawing and it will be visible if the bounding box is adjusted) and the drawing may not fit on the standard A4 paper size. The *NoClip* export adjusts the size of the drawing so it can be printed on A4 paper (portrait), and the *Clip* file adjusts the boundaries of the drawing so all vertices and labels are included and there is more or less the same white margin everywhere. We recommend the *Clip* format.

Most Windows applications cannot show Encapsulated PostScript figures, so you have to use a special viewer or translate the drawing into another format. GhostScript and GhostView software is a free viewer for (Encapsulated) PostScript, which can be downloaded from <http://www.cs.wisc.edu/~ghost/>. Install both programs on your computer and you can view (Encapsulated) PostScript drawings by opening the .eps or .ps file (*File>Open* command) in GhostView. From GhostView, you can print the drawing on any printer and export it in several graphical formats (including Windows MetaFile if you also install `pstoedit.exe`, which is freely available at <http://www.pstoedit.net/>). With Acrobat Distiller, which is part of the Adobe Acrobat software package (not free; check <http://www.adobe.com/products/acrobat/main.html>), you can translate Encapsulated PostScript to a Portable Document Format (PDF), which can be viewed with the free Adobe Acrobat Reader (<http://www.adobe.com/products/acrobat/readstep2.html>).

Pajek offers many options for customizing its Encapsulated PostScript export, which we present in Section A2.2. For onscreen editing Encapsulated PostScript drawings, however, you need dedicated drawing software such as CorelDraw. In CorelDraw, you can import Encapsulated PostScript files (*File>Import* command in CorelDraw) and translate it to the vector format of CorelDraw (select the file type “PostScript Interpreted PS,PRN,EPS”). If you ungroup the imported drawing, you can change each element individually. Thus, you can completely customize the sociogram. In Figure 10 in Chapter 2, for instance, we combined the sociogram of world trade in miscellaneous manufactures of metal with a line drawing of the continents (available as `outline.wmf`). In addition, drawing software such as CorelDraw can export the drawing to a format that can be embedded and viewed in most Windows text processors. We advise using the Windows MetaFile (WMF) format, which is also vector based.

A2.1.3 Scalable Vector Graphics

What PostScript is to printing Scalable Vector Graphics (SVG) is to publishing on the Web. Just as with PostScript, SVG defines each distinct shape and assembles the drawing at the moment it is viewed, so it produces optimal quality in every resolution. In addition, SVG pictures can be changed interactively on the Web. As a consequence, there are several commands for exporting SVG in Pajek's *Export>SVG* submenu, each of which adds a specific type of interaction to the drawing.

The most basic way of exporting a network as SVG is provided by the *Export>SVG>General* command, which produces an image of the network as it is shown in the Draw screen, without interactivity. In the *Save As* dialog box, enter a filename, which will be used for both the SVG file and the HTML file (Web page) within which the drawing will be displayed. Note that you need both files (in the same directory) to view the drawing. The second SVG export command, *Labels/Arcs/Edges*, exports the drawing with checkboxes, which enable the user to toggle the display of arcs, edges, arc labels, edge labels, and vertex labels.

When the network is drawn with a partition, the *Export>SVG>Partition* submenu contains options for interactively manipulating the classes of the partition. The *Classes* command produces checkboxes with which vertices belonging to a class can be hidden (or displayed), including the lines among them. The lines between different classes can be hidden separately. The *Classes with semi-lines* command produces similar output but lines incident with vertices that are hidden are changed to semi-lines (half lines) and semi-lines are hidden on removal of the vertices with which they are incident. The *Nested Classes* command adds a checkbox for each class in the partition but toggling one checkbox automatically toggles the value of all lower classes. This is very useful for displaying nested classes, such as *m*-slices (Chapter 5) and *k*-cores (Chapter 3). Note that it is possible to display two partitions in the SVG export. If two partitions are selected in the *Partitions* menu of the Main screen, the first partition defines the classes that can be selected by means of the checkboxes, whereas the second partition determines the colors of the vertices in the SVG drawing.

If the network contains lines with different values, the SVG export can create checkboxes for showing or hiding lines with particular values. The *Export>SVG>Line Values* submenu contains commands for exporting ordinary classes (*Classes*) and nested classes (*Nested Classes*). Line values must be grouped into classes, so a dialog box appears asking for the number of classes or the class boundaries. In addition, the submenu contains options for the appearance of lines (*Export>SVG>Line Values>Options*), which can receive different colors (*Different Colors* option) or grays (*GrayScale* option), or the line width can be made to represent the class of line values (*Different Widths* option).

Finally, it is possible to create a series of SVG images, all linked by previous and next buttons, for a list of networks, partitions, or vectors. First, make sure that the right data objects are selected in the *Options>Previous/Next>Apply to* submenu of the Draw screen. If you have only a sequence of networks to display, the *Network* option should

Export>SVG

*Export>SVG>
General*

*Export>SVG>
Labels/Arcs/
Edges*

*Export>SVG>
Partition>
Classes*

*Export>SVG>
Partition>
Classes with
Semi-Lines*

*Export>SVG>
Partition>
Nested Classes*

*Export>SVG>
Line Values>
Classes*

*Export>SVG>
Line Values>
Nested Classes*

*[Draw]Options>
Previous/Next>
Apply to*

*Export>SVG>
Current and all
Subsequent*

be selected only, but if you have a sequence of networks and a sequence of partitions, both *Network* and *Partition* should be selected. Then select the first network, partition, or vector that you want to export in the drop-down menus of the Main screen, draw the network and, if applicable, partition and vector. Finally, select the *Export>SVG>Current and all Subsequent* option and execute one of the *Export>SVG* commands discussed above.

SVG drawings are viewed in an Internet browser. As mentioned, the SVG drawing is inserted in an HTML page, which is read by a Web browser. Recent browsers such as Microsoft Internet Explorer version 6.0 and higher can display SVG images directly. For older browsers, you need to install a plug in, which can be downloaded from <http://www.adobe.com/svg/viewer/install/>.

Onscreen editing of SVG images can be done with several software packages, among them Webdraw (<http://www.jasc.com/products/webdraw/>) and Adobe Illustrator (<http://www.adobe.com/products/illustrator/main.html>).

A2.1.4 Virtual Reality Modeling Language

*Layout>Energy
>Fruchterman
Reingold>3D*

Virtual Reality Modeling Language (VRML) is a standard language for defining three-dimensional objects that can be published on the Web. In a Web browser, VRML models (e.g., a three-dimensional model of a social network) can be rotated, walked through, and so on. You may optimize the network's layout in three dimensions with the *Layout>Energy>Fruchterman Reingold>3D* command before you export a VRML model from Pajek.

Export>VRML

With the *Export>VRML* command, Pajek exports the network to VRML 1.0. A dialog box prompts for a file name in which to store the model. We advise using the default extension *.wrl*, which stands for world, so Internet browsers can correctly identify the file type. If the network was drawn with a partition, the spheres in the model, which represent the vertices, have the right colors. Vertex labels will be transformed to anchor names.

For displaying a VRML world in a Web browser, you need a plug in, for instance, Cosmo player (<http://www.ca.com/cosmo/home.htm>) or Cortona VRML Client (<http://www.parallelgraphics.com/products/downloads>). Some plug-ins can read only newer VRML models (VRML 2.0 or 97). If so, you need a converter such as the Cortona VRML 1.0 converter or the MS-DOS program *VRML1TOVRML2.EXE*, which must be run from the MS-DOS prompt with the command *VRML1TOVRML2* followed by the names of the VRML 1.0 file and the new VRML 2.0 file separated by spaces. Maybe the original vertex names, which are anchor names in the VRML model, may cause problems; if this happens, open the the VRML file in a plain text editor and adjust or remove the anchor names. Do not forget to save the changed file to plain text.

Some layout properties of VRML models can be set in Pajek (see Section A2.2). If you want to edit a VRML model onscreen, you need special

software. Consult <http://www.web3d.org/x3d/vrml/tools/authoring/> for VRML design and editing software.

If you want to include an image of a three-dimensional model in a document, it is a good idea to ray-trace it. Ray-tracing is a technique that renders very realistic images from three-dimensional models by calculating (tracing) the trajectory of light from the virtual light sources and the reflection of this light on the surfaces in the model. In Chapter 1 Figure 14 provides an example. POVRay is a good and free software package for ray-tracing (<http://www.povray.org/>). However, it does not understand the Virtual Reality Modeling language directly, so you must first transform the VRML 2.0 model to the POVRay language, which can be done with the MS-DOS program `vrml2pov.exe`. In the MS-DOS prompt, issue the command `vrml2pov` followed by the names of the VRML 2.0 file and the new POVRay file, separated by spaces. For the POVRay file, use the extension `.pov`. You can open the newly created file in POVRay and render it. The rendered image is automatically saved in the format specified in POVRay's INI file. Enjoy, but realize that it may take quite some time before you master the software and achieve satisfactory results.

A2.1.5 MDL MOL and Kinemages

Molecular structures can be modeled as networks, so the software developed by scientists to visualize molecules can be used for the display of social networks. Pajek can export to two molecule model formats: MDL MOL and Kinemages. MDL MOL models are three-dimensional and they are viewed in a Web browser using a special plug-in (Chime, which can be downloaded from <http://www.mdli.com/>). This format does not offer real advantages over VRML, so we do not discuss it in detail.

*Export>MDL
MOLfile*

In contrast, the other format – Kinemages – does have special features that need to be mentioned. This format is designed to present sequences of three-dimensional images. In principle, this can also be done in VRML but in Kinemages, it is very easy to add text to each image and to include questions. This software is excellent for teaching purposes and for electronic publishing in the case of undirected networks. For directed or signed networks, Kinemage is less useful because it does not display the direction of arcs or the signs of lines.

Pajek can produce these sequences with the *Export>Kinemages>Current and all Subsequent* command. Just like SVG export, you must select the data objects (network, partition, vector) that you want to export in the *Options>Previous/Next>Apply to* submenu of the Draw screen and you must select the first network (partition, vector) that you want to export in the droplists of the Main screen. With the *Export>Kinemages>Current and all Subsequent* command, Pajek creates one Kinemage file (default extension is `.kin`) containing the series of networks. In the *Save as Type* selection box of the *Save As* dialog box, you can choose whether you want the vertices represented by spheres or by vertex labels in the Kinemage.

*Export>
Kinemages>
Current and all
Subsequent*

*[Draw]
Options>
Previous/Next>
Apply to*

You need special software for viewing Kinemages, which is called Mage (<http://kinemage.biochem.duke.edu/kinemage/kinemage.php>). Open the

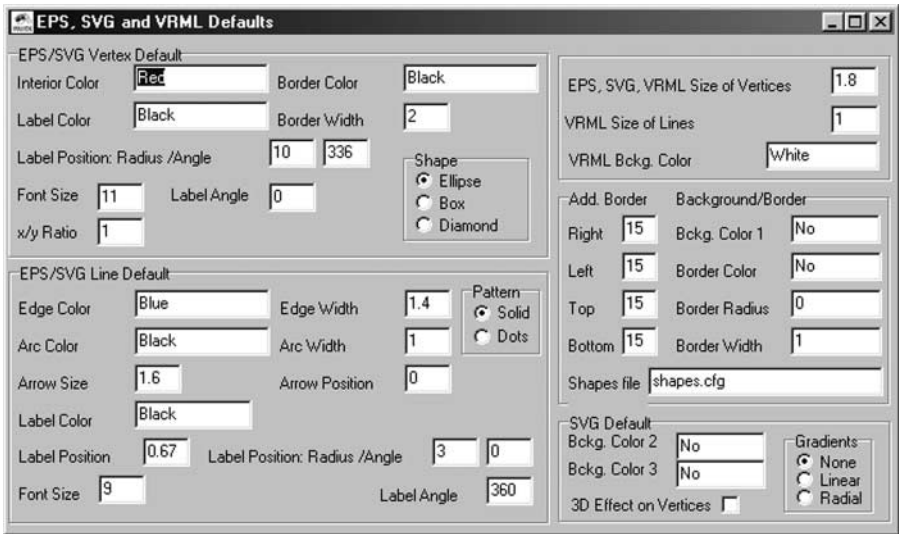


Figure 141. The *Options* screen.

Kinimage created by Pajek in Mage and step through the sequence of images with the *KINEMAGE/Next* (or *Ctrl-n*) command in Mage.

A2.2 Layout Options

[Draw]
Export>Options

PostScript, SVG, and VRML visualizations need additional information about the size, placement, and colors of vertices, lines, labels, and background. This information can be supplied either with the *Options* command in the *Export* menu of the Draw screen or in the network data file. The *Export>Options* (Figure 141) define layout properties of all vertices and lines, whereas the parameters in the network data file define the layout of each individual vertex or line. Layout parameters specified in the network data file override the settings in the *Export>Options dialog screen*. If, for instance, the color of vertices is specified in the network data file, entering a color in the *Options* window has no effect.

The *Export>Options* command opens a window that is divided into five frames, two on the left and three on the right (Figure 141). We discuss each frame separately and, where appropriate, we mention the layout parameter that defines the layout feature in the network data file. Note that you should type the layout parameters at the end of the line defining a vertex, edge, or arc in the network data file and you must separate the parameters and their values by one or more blanks. Consult the document *DrawEPS.htm* (on the book's Web site) for a description of all layout parameters in the network data file.

[Main] Options>
Ini File>Save
[Main] Options>
Ini File>Load

If you want to save the settings of a particular layout, use the *Options>Ini File>Save* command in the Main screen. This command saves all present settings to a file, which can be loaded (command *Options>Ini File>Load*) to restore the settings.

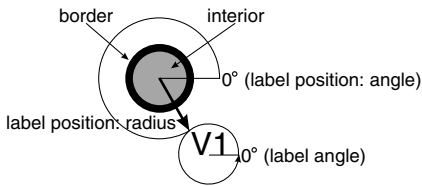


Figure 142. Layout of a vertex and its label.

A2.2.1 Top Frame on the Left – EPS/SVG Vertex Default

This frame defines how vertices are drawn when you export 2D layouts to (Encapsulated) PostScript and Scalable Vector Graphics. Figure 142 shows some important properties of the layout of a vertex and its label.

In the top frame, the color of the interior (*Interior Color*) and border (*Border Color*) of the vertices can be specified. Enter one of the colors that are listed in Table 26 (the colors are shown in the file *Crayola.pdf*, which is included in the documentation accompanying Pajek) and make sure you use upper- and lowercase letters as specified there. The width of the border (*Border Width*; parameter *bw* followed by a number in the network data file) and the shape of the vertex (*Shape*; just type *ellipse*, *box*, *diamond*, *cross*, or *empty* in the network data file – this parameter must precede all other layout parameters except for the vertex’s coordinates) can be defined. The user may choose among the predefined shapes ellipse, box, and diamond and he or she may squeeze or stretch the shape horizontally by adjusting the *x/y* ratio (*x/y Ratio*; parameter *x_fact* followed by a number). If this ratio is smaller than 1, the shape of the vertex is squeezed; if it is larger than 1, it is stretched. In Figure 143, the horizontal diameter of the vertex on the left is half the vertical diameter,

[Draw] Export>
Options>
Interior Color

[Draw] Export>
Options>
Border Color

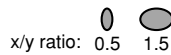
[Draw] Export>
Options>
Border Width

[Draw] Export>
Options>
Shape

[Draw] Export>
Options>
x/y
ratio

Table 26. *Names of Colors in Pajek*

Apricot	Fuchsia	MidnightBlue	RoyalPurple
Aquamarine	Goldenrod	Mulberry	RubineRed
Bittersweet	Gray	NavyBlue	Salmon
Black	Green	OliveGreen	SeaGreen
Blue	GreenYellow	Orange	Sepia
BlueGreen	JungleGreen	OrangeRed	SkyBlue
BlueViolet	Lavender	Orchid	SpringGreen
BrickRed	LFadedGreen	Peach	Tan
Brown	LightCyan	Periwinkle	TealBlue
BurntOrange	LightGreen	PineGreen	Thistle
CadetBlue	LightMagenta	Pink	Turquoise
Canary	LightOrange	Plum	Violet
CarnationPink	LightPurple	ProcessBlue	VioletRed
Cerulean	LightYellow	Purple	White
CornflowerBlue	LimeGreen	RawSienna	WildStrawberry
Cyan	LSkyBlue	Red	Yellow
Dandelion	Magenta	RedOrange	YellowGreen
DarkOrchid	Mahogany	RedViolet	YellowOrange
Emerald	Maroon	Rhodamine	
ForestGreen	Melon	RoyalBlue	

Figure 143. The x/y ratio of a vertex.

so its x/y ratio is 0.5. The horizontal diameter of the other vertex is 50 percent longer than the vertical diameter.

[Draw] Export>
Options>Label
Color

The color (*Label Color*; parameter `lc` followed by a color name) and font size (*Font Size*; parameter `fos` followed by a number) of the label can also be adjusted in this frame. Again, use names of colors as listed in Table 26. In addition, the orientation of the vertex label (*Label Angle*; `la` followed by a number indicating the degree) can be changed relative to a horizontal line. Zero degrees is usually the best choice because this will display the labels horizontally. Label angles from 360 to 720 degrees position the label relative to the center of the layout, which can be useful when all vertices are drawn in concentric circles.

[Draw] Export>
Options>Font
Size

[Draw] Export>
Options>Label
Angle

[Draw] Export>
Options>Label
Position:
Radius/Angle

The most complicated part, however, is the position of the label with respect to the vertex. The position of the label is defined by its distance to the center of the vertex (the radius) and by the angle from the horizontal line starting at this center, ranging from 0 to 360 degrees (*Label Position: Radius/Angle*; parameters `lr` followed by a number and `lphi` followed by a degree respectively). If the angle of the label position is zero, the label is resting on this horizontal line at the right of the vertex. If the radius is zero, the label is placed in the center of the vertex. With short labels and large vertices (see Section A2.2.3) or a high x/y ratio, the label may fit inside the vertex. To place labels outside but near their vertices, enter a positive number in the *Label Position: Radius* field. It is difficult to give a general rule for a good size of the radius, because it depends on the size of the vertex and the size of the label's type font. Most figures of this book used the sizes shown in Figure 141.

A2.2.2 Bottom Frame on the Left – EPS/SVG Line Default

This frame defines the way the lines are drawn when 2D layouts are exported to (Encapsulated) PostScript and Scalable Vector Graphics. It contains fields defining the appearance of lines and line values or line labels as well as fields that specify the orientation of labels and their location with respect to the lines to which they belong.

[Draw] Export>
Options>Edge
Color, Arc Color,
Label Color,
Pattern, Edge
Width, Arc
Width, Arrow
Size, Font Size,
Arrow Position

The colors of edges and arcs (*Edge Color* and *Arc Color*; parameter `c` followed by a color name) and labels (*Label Color*; `lc` followed by a color name) can be changed by entering names of colors as they appear in Table 26. Note that arcs and edges may receive different colors. You may draw all lines as dotted lines by selecting *Dots* in the *Pattern* field (parameter `p` followed by `Solid` or `Dots`). The width of edges and arcs (*Edge Width* and *Arc Width*; `w` followed by a number) may also differ; it is good practice to draw edges a bit wider than arcs. The size of the arrowhead (*Arrow Size*; `s` followed by a number) is specified independently of the arc's width. The size of the labels is defined by their font size (*Fontsize*; `fos` followed by a number), which should be smaller than the font size of the vertex labels for pleasant results.

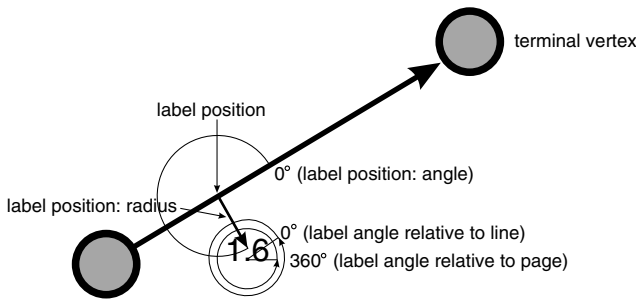


Figure 144. The position and orientation of a line label.

You may decide where to place the arrowhead on the line (*Arrow Position*; `ap` followed by a number). A number between 0 and 1 is interpreted as a proportion expressing the relative distance from the end to the beginning of the arc; for example, 0 will place the arrowhead at the end of the arc near the terminal vertex and 0.5 will situate the arrowhead in the middle of the arc. A distance larger than 1 is interpreted as an absolute distance from the terminal vertex. This is useful if you want to have all arcs on the same distance from the terminal vertex, regardless of the arc's length. You have to experiment a little to find a nice distance.

The remaining fields concern the orientation and position of the line label (e.g., its value) with respect to the line. Figure 144 illustrates the relevant parameters. First, choose the position on the line (edge or arc) to which the line label is attached (*Label Position*; `lp` followed by a number). This is similar to the position of the arrowhead: numbers between 0 and 1 are proportions, that is, relative distances from the end to the beginning of the line, and numbers above 1 are absolute distances from the end of the line. In Figure 144, the label position is 0.67, so it is located at two-thirds of the arc, measured from its end. Note that edges do not have a start and an end, so it is most appropriate to position the edge labels halfway.

[Draw] Export>
Options>Label
Position

Next, the location of the center of the label with respect to the position on the line is defined by two properties that are similar to the location of vertex labels: radius and angle. The radius (*Label Position: Radius*; `lr` followed by a number) is the distance between the position on the line and the center of the line label that is measured at the specified angle (*Label Position: Angle*; `lphi` followed by a number of degrees) from the line. In Figure 144, the angle is 270 degrees.

[Draw] Export>
Options>Label
Position: Radius

[Draw] Export>
Options>Label
Position: Angle

Finally, the orientation of the line label is defined in the *Label Angle* field (parameter `la` followed by a number of degrees). An angle smaller than 360 degrees is measured relative to the direction of the line, where zero degrees is parallel to the line. Angles of 360 degrees and more are relative to a horizontal line. For easy reading, an angle of 360 degrees is optimal because it displays line labels horizontally.

[Draw] Export>
Options>Label
Angle

A2.2.3 Top Frame on the Right

This frame defines additional defaults when we export layouts to Virtual Reality Modeling Language (VRML). The field *EPS*, *SVG*, *VRML Size of*

[Draw] Export>
Options>EPS,

SVG, VRML
Size of Vertices

[Draw] Export>
Options>
VRML Size of
Lines

[Draw] Export>
Options>VRML
Bckg. Color

Vertices (parameter *s_size* followed by a number) specifies the default size of vertices when exporting to VRML but also to EPS and SVG. The diameter of lines in VRML can be changed in the *VRML Size of Lines* (no layout parameter for the network data file) field. It is difficult to give general rules about optimal settings for these fields; you have to experiment.

In the *VRML Bckg. Color* field (no layout parameter for the network data file), you can choose a background color for the layout in VRML: just enter one of the color names listed in Table 26.

A2.2.4 Middle Frame on the Right

[Draw] Export>
Options>
Add. Border

This frame defines additional properties of the network layout when we export layouts to Encapsulated PostScript and Scalable Vector Graphic (there are no layout parameters for the network data file). Most fields relate to a border around the layout. With the four fields *Right*, *Left*, *Top*, and *Bottom*, you can add additional space to the right, left, and so on of the picture. This is effective for SVG exports and for EPS exports when the EPS Clip format is selected.

[Draw] Export>
Options>
Border Color

The color of the border (*Border Color*) as well as the background (*Bckg. Color*) can be picked from the color names listed in Table 26. Enter the word *No* in these fields if you do not want a border or background color.

[Draw] Export>
Options>Bckg.
Color

The shape of the borderline is characterized by the fields *Border Radius* and *Border Width*. If the radius of the border is zero, the border is a rectangle. Higher values (e.g., 10, 50, or 100) round off the corners. A border width of one unit produces a rather thin borderline and higher values yield fatter borderlines.

[Draw] Export>
Options>
Border Radius

Finally, this window contains a field in which a file can be selected with custom shapes for vertices. The default is the file *shapes.cfg*. Defining your own custom shapes demands knowledge of the PostScript language.

[Draw] Export>
Options>
Border Width

A2.2.5 Bottom Frame on the Right – SVG Default

[Draw] Export>
Options>Bckg.
Color 2

The last frame contains additional options for exporting Scalable Vector Graphics (again, there are no layout parameters for the network data file).

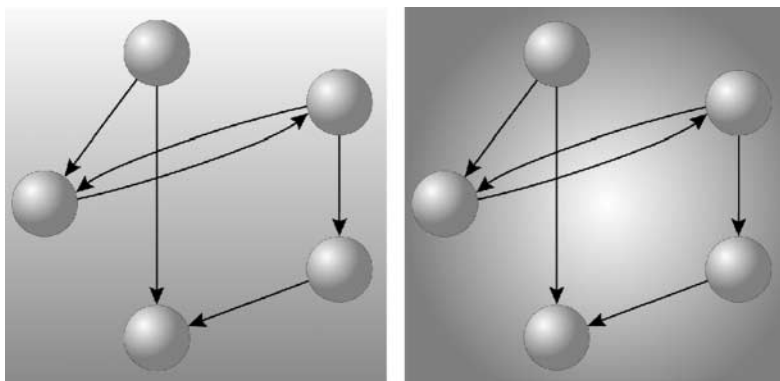


Figure 145. Gradients in SVG export: linear (left) and radial (right).

These layouts can handle gradient colors: an area in which one color gradually blends into another color. You can specify three colors. The first color is the background color defined in the middle frame (see Section A2.2.4). The second color can be selected in the *Bckg. Color 2* field. Use the color names of Table 26 and enter the word *No* if you do not want to use a second color and hence no gradient. If you like, you can select a third color in the *Bckg. Color 3* field. The type of blend can be selected in the *Gradients* field: *Linear* or *Radial* (see Figure 141). In a linear gradient, the original background color is at the top of the area and blends into the second and third colors on its way down. In a radial gradient the middle of the drawing is colored with the original background color. Figure 145 also shows the effects of the *3D Effect on Vertices* option.

[Draw] Export>
Options>Bckg.
Color 3

[Draw] Export>
Options>
Gradients

[Draw] Export>
Options>3D
Effect on
Vertices