Melissa Clarkson
March 2010
sna version 2.0-1
document version 2

A guide to the gplot function of the sna library for R

This function creates a node-and-edge visualization of a network. It can accept an adjacency matrix (N×N), an array of adjacency matrices (m×N×N), `network` objects, and other forms of network data described in the `sna` library documentation.

The form of the function

`gplot(data_matrix)` will draw a graph using the default parameters.

`gplot(data_matrix, argument=parameter)` is used to specify parameters.

The data for these examples

The graph in this example has only two nodes, with a directed relationship from the first to the second node. There is also a loop on the first node, which will be used in some examples.

This matrix indicates presence of an edge with "1", absence with "0".

```
data_matrix <- matrix(c(1, 0, 1, 0), nrow=2)
```

```
     [,1] [,2]
[1,]    1    1
[2,]    0    0
```

This matrix has edge values.

```
data_valued_matrix <- matrix(c(0.8, 0, 0.3, 0), nrow=2)
```

```
     [,1] [,2]
[1,] 0.8  0.3
[2,] 0.0  0.0
```

The positions of each node within the plot can also be specified in a two-column matrix. Each row represents a node, with the x-coordinate in the first column and the y-coordinate in the second column.

```
position_matrix <- matrix(c(0, 3, 0, 0), nrow=2)
```

```
     [,1] [,2]
[1,]    0    0
[2,]    3    0
```

This is a vector of the labels for the nodes

```
label_vector <- c("first label", "second label")
```

```
[1] "first label"  "second label"
```

Arguments and parameters

Selecting a graph from a set

g	integer	if a set of graphs was provided as data (such as an array of adjacency matrices), this is the index of the graph to be drawn (default = 1)

Type of graph

`gmode`	`"digraph"`	interpret edges as directed (default)
	`"graph"`	interpret edges as undirected
	`"twomode"`	rows and columns of data matrix are distinct nodes
`diag`	`FALSE`	do not treat the diagonal of the matrix as data (default)
	`TRUE`	the diagonal is data (creates loops)

```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, diag=FALSE)
```

```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, diag=TRUE)
```

What is to be drawn

`new`	`TRUE`	a new graph will be drawn (default)
	`FALSE`	nodes and edges will be added to existing graph
`thresh`	num. value	by default, all edges with value 0 or greater are drawn; if a value is specified, only edges above this value will be drawn

```
gplot(data_valued_matrix, coord=position_matrix, jitter=FALSE, thresh=0.2)
```

```
gplot(data_valued_matrix, coord=position_matrix, jitter=FALSE, thresh=0.9)
```

`displayisolates`	`TRUE`	display nodes that have no edges (default)
	`FALSE`	do not display nodes that have no edges

Positions of the nodes

`coord`	two-column matrix	to specify the position of each node, provide a matrix (one row per node; first column with x-coordinate, second column with y-coordinates
`mode`	`"fruchtermanreingold"` is the default node layout method	

see documentation on gplot layout algorithms for other layout modes and how to specify parameters

`jitter`	`TRUE`	jitter will be added to positions of nodes determined by either the layout algorithm or the matrix of node coordinates (default)
	`FALSE`	no jitter

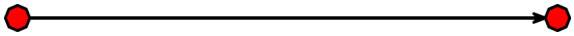
Node-mark attributes

`vertex.col`	color	node color (default = red, from second position of the default color palette; for two-mode data the second node set is blue, from the fourth position); use a vector if nodes are to have different colors *see documentation on specifying colors*

```
gplot(data_matrix, coord=position_matrix, jitter=FALSE,
vertex.col=rgb(68,135,203, maxColorValue=255))
```

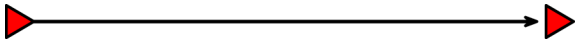
`vertex.cex`	num.value	expansion factor for nodes (default = 1); use a vector if nodes are to have different sizes

```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, vertex.cex=1)
```

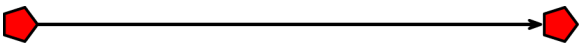


```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, vertex.cex=2)
```

`vertex.sides`	integer	number of sides for node polygons (default = 8); use a vector if nodes are to have different numbers of sides

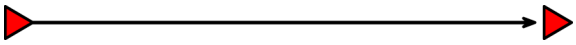


```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, vertex.cex=3,
vertex.sides=3)
```

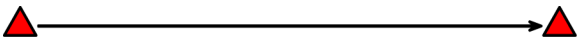


```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, vertex.cex=3,
vertex.sides=5)
```

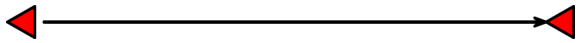
`vertex.rot`	num. value	angle for rotation of nodes (default = 0); use a vector if nodes are to have different degrees of rotation



```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, vertex.cex=3,
vertex.sides=3, vertex.rot=0)
```



```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, vertex.cex=3,
vertex.sides=3, vertex.rot=90)
```



```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, vertex.cex=3,
vertex.sides=3, vertex.rot=180)
```

`vertex.border` — color

color for border of nodes (default = black, from first position of default color palette);
use a vector if nodes are to have different border colors

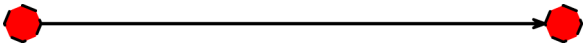
see documentation on specifying colors

```
gplot(data_matrix, coord=position_matrix, jitter=FALSE,
vertex.border=rgb(68,135,203, maxColorValue=255))
```

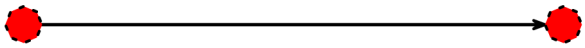
`vertex.lty` — line type

line type for border of nodes (default = solid line);
use a vector if nodes are to have different border types

see documentation on line styles



```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, vertex.cex=3,
vertex.lty=2)
```



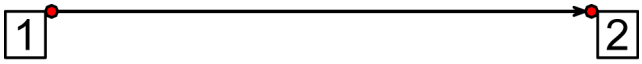
```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, vertex.cex=3,
vertex.lty=3)
```

Node labels

`displaylabels`

FALSE — do not display labels for nodes;
by default, will display labels only if `label` is given a vector of label names

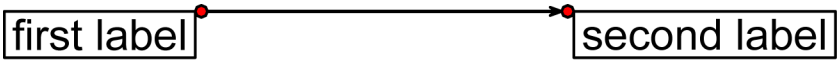
TRUE — display labels for nodes;
the numbers or labels in the dataset will be used unless `label` is given
a vector of label names



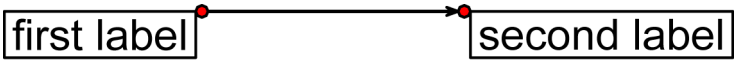

```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, displaylabels=TRUE)
```

`label` — vector

a vector of labels for the nodes



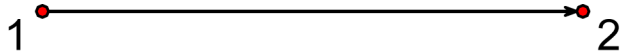

```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, label=label_vector, pad=1)
```



pad plotting range if labels are being clipped

```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, label=label_vector, pad=2)
```

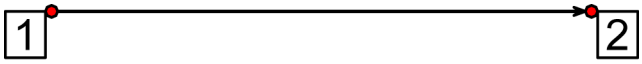
`boxed.labels`	TRUE	draw boxes around node labels (default)
	FALSE	do not draw boxes on node labels




```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, displaylabels=TRUE,
boxed.labels=FALSE)
```

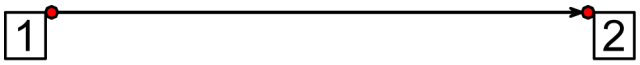
`label.lwd`	num. value	line width for node label boxes (default = 1); use a vector if boxes are to have different border widths *see documentation on line styles*


```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, displaylabels=TRUE,
label.lwd=0.5)
```

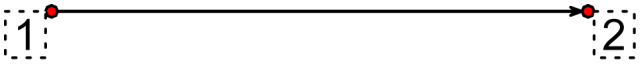



```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, displaylabels=TRUE,
label.lwd=1)
```

`label.lty`	line type	line type for border of node label boxes (default = solid line); use a vector if boxes are to have different border line types *see documentation on line styles*

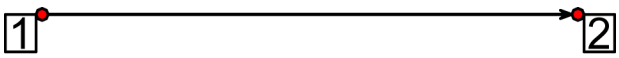



```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, displaylabels=TRUE,
label.lty=1))
```

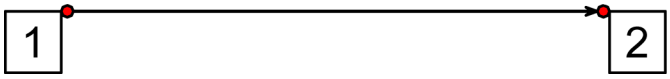



```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, displaylabels=TRUE,
label.lty=3)
```

`label.pad`	num.value	amount of space between node label and border of box (default = 0.5)




```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, displaylabels=TRUE,
label.pad=0.2
```

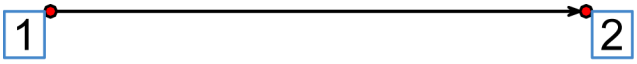



```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, displaylabels=TRUE,
label.pad=1
```

`label.border` color

color for border of node label boxes (default = black, from first position of default color palette);
use a vector if box borders are to have different colors

see documentation on specifying colors

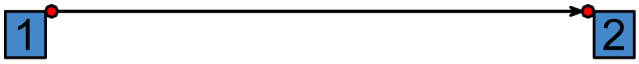



```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, displaylabels=TRUE,
label.border=rgb(68,135,203, maxColorValue=255))
```

`label.bg` color

color for background of node label boxes (default = white);
use a vector if boxes are to have different background colors

see documentation on specifying colors

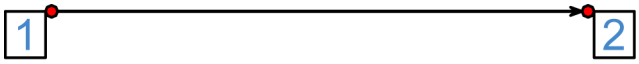



```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, displaylabels=TRUE,
label.bg=rgb(68,135,203, maxColorValue=255))
```

`label.col` color

color for text of node labels (default = black, from first position of default color palette);
use a vector if labels are to have different colors

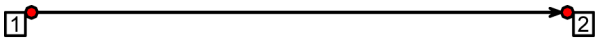
see documentation on specifying colors




```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, displaylabels=TRUE,
label.col=rgb(68,135,203, maxColorValue=255))
```

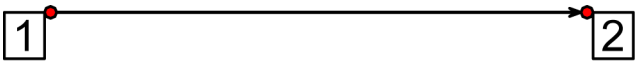
`label.cex` num. value

expansion factor for node labels (default = 1)

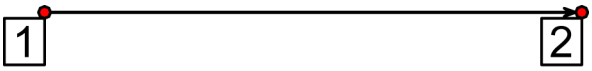



```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, displaylabels=TRUE,
label.cex=0.5)
```

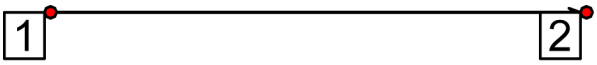
`label.pos` — position of labels; (default = 0, which positioins labels away from center of graph)



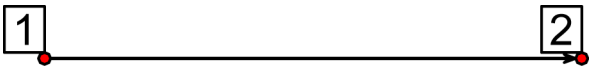
0 — `gplot(data_matrix, coord=position_matrix, jitter=FALSE, displaylabels=TRUE, label.pos=0)`



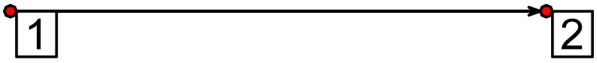
1 — `gplot(data_matrix, coord=position_matrix, jitter=FALSE, displaylabels=TRUE, label.pos=1)`



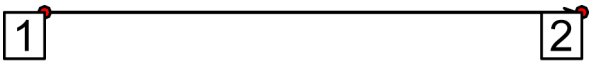
2 — `gplot(data_matrix, coord=position_matrix, jitter=FALSE, displaylabels=TRUE, label.pos=2)`



3 — `gplot(data_matrix, coord=position_matrix, jitter=FALSE, displaylabels=TRUE, label.pos=3)`



4 — `gplot(data_matrix, coord=position_matrix, jitter=FALSE, displaylabels=TRUE, label.pos=4)`



5 — `gplot(data_matrix, coord=position_matrix, jitter=FALSE, displaylabels=TRUE, label.pos=5)`

Edge-mark attributes

Note: edges are drawn as polygons, so they have both a border and fill.
Methods are not provided in gplot to control the fll independently of the border.

`usearrows`

TRUE — draw arrows on edges

FALSE — do not draw arrows on edges

`gplot(data_matrix, coord=position_matrix, jitter=FALSE, usearrows=TRUE)`

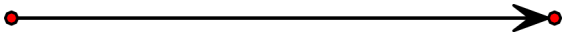
`gplot(data_matrix, coord=position_matrix, jitter=FALSE, usearrows=FALSE)`

`arrowhead.cex` num. value

expansion factor for arrowheads (default = 1)
use a vector if arrowheads are to have different sizes

```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, arrowhead.cex=1)
```

```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, arrowhead.cex=1.5)
```



```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, arrowhead.cex=3)
```

`edge.col` color

color for edge border and fill (default = black, first position of default color palette);
use a vector if edges are to have different colors

see documentation on specifying colors

```
gplot(data_matrix, coord=position_matrix, jitter=FALSE,
edge.col=rgb(68,135,203, maxColorValue=255))
```

`edge.lwd` num. value

thickness of edges (default = 0); controls distance between borders of edge;
use a vector or adjacency matrix if edges are to have different thicknesses

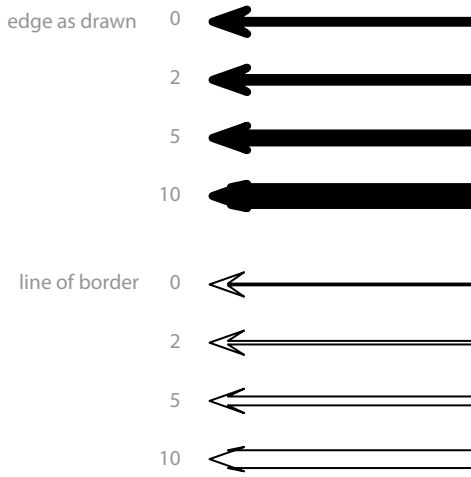
```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, edge.lwd=0)
```

```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, edge.lwd=2)
```

```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, edge.lwd=5)
```

Edge width is automatically scaled by the data values in the matrix. The only way to display all edges of a valued matrix with the same width is to leave `edge.lwd` at the default value of zero.

Notice that this function alters only the width of the edge, not the width of the arrow head. The scaling factor for the arrow head can be set separately.

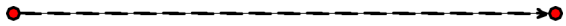


`edge.lty` line type

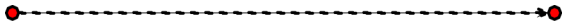
line type for border of edges (default = solid line);
use a vector if edges are to have different border line types

see documentation on line styles

```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, edge.lty=1)
```



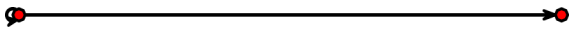
```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, edge.lty=2)
```



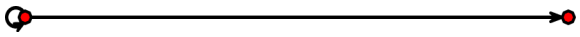
```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, edge.lty=3)
```

`loop.cex` num. value

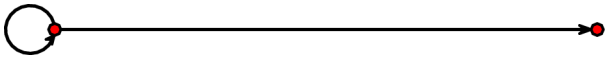
expansion factor for loops (default = 1);
use a vector if loops are to have different sizes



```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, diag=TRUE, loop.cex=1)
```



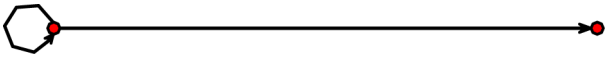
```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, diag=TRUE, loop.cex=1.5)
```



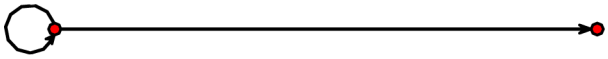
```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, diag=TRUE, loop.cex=4)
```

`loop.steps` integer

number of segnements to use when drawing loops (default = 20)

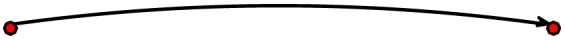


```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, diag=TRUE, loop.cex=4,
loop.steps=6)
```



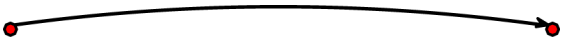
```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, diag=TRUE, loop.cex=4,
loop.steps=10)
```

usecurve	FALSE	draw straight edges (default)
	TRUE	draw curved edges



```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, usecurve=TRUE)
```

edge.curve	num. value	value to control amount of curvature in edges (default = 0.1); use a vector or adjacency matrix if edges are to have different curvatures



```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, usecurve=TRUE,
edge.curve=0.1)
```

```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, usecurve=TRUE,
edge.curve=0.5)
```

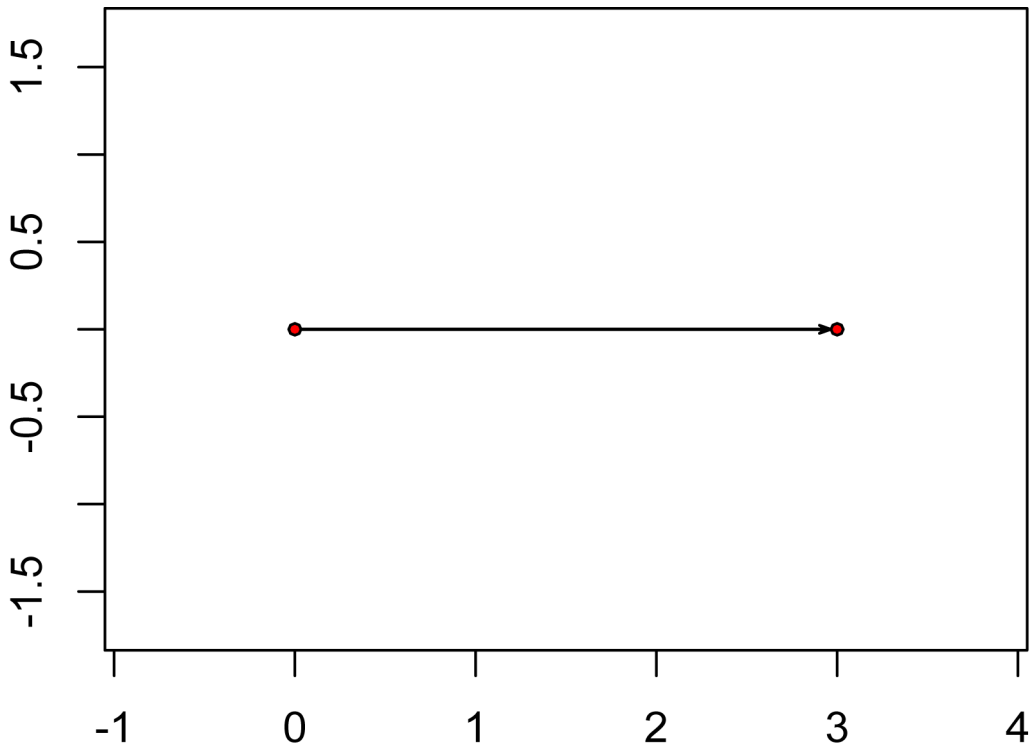
edge.steps	integer	number of segnements to use when drawing curved edges (but not loops) (default = 50)

```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, usecurve=TRUE,
edge.curve=0.5, edge.steps=5)
```

```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, usecurve=TRUE,
edge.curve=0.5, edge.steps=15)
```

Other arguments

`suppress.axes`	TRUE	do not draw axes on plot (default)
	FALSE	draw axes on plot; parameters to control the axes are available in `plot()`, and include `xlim` and `ylim` for axis limits, `xlab` and `ylab` for axis labels




```
gplot(data_matrix, coord=position_matrix, jitter=FALSE, suppress.axes=FALSE)
```

`interactive`	FALSE	draw graph without interactive node placement (default)
	TRUE	allow for interactive node placement
`interact.bycomp`	FALSE	during interactive node placement, only single node will be moved (default)
	TRUE	during interactive node placement, entire component will be moved
`object.scale`	num. value	base length for plotting objects (default = 0.01); defined as a fraction of the linear scale of the plotting region
`vertices.last`	TRUE	draw nodes after edges
	FALSE	draw nodes before edges

Examples of assigning node-mark and edge-mark attributes from data values

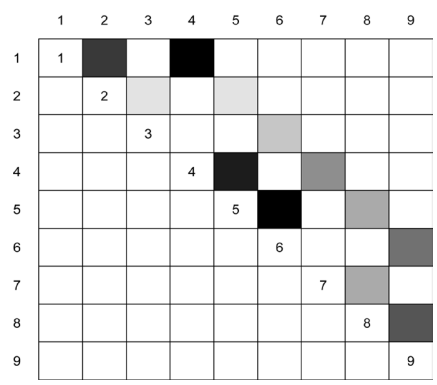
This example demonstrates how to set node-mark and edge-mark attributes for individual nodes and edges, using data values.

The data for these examples

The graph for these examples has nine nodes and twelve directed edges.

Adjacency matrix with edge values:

```
data_vector <- c(0,0,0,0,0,0,0,0,0, 0.7,0,0,0,0,0,0,0,0,
0,0.1,0,0,0,0,0,0,0, 0.9,0,0,0,0,0,0,0,0, 0,0.1,0,0.8,0,0,0,0,0,
0,0,0.2,0,0.9,0,0,0,0, 0,0,0,0.4,0,0,0,0,0, 0,0,0,0,0.3,0,0.3,0,0,
0,0,0,0,0,0.5,0,0.6,0)

data_matrix <- matrix(data_vector, nrow=9)
```




```
plot.sociomatrix(data_matrix)
```

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]	[,7]	[,8]	[,9]
[1,]	0	0.7	0.0	0.9	0.0	0.0	0.0	0.0	0.0
[2,]	0	0.0	0.1	0.0	0.1	0.0	0.0	0.0	0.0
[3,]	0	0.0	0.0	0.0	0.0	0.2	0.0	0.0	0.0
[4,]	0	0.0	0.0	0.0	0.8	0.0	0.4	0.0	0.0
[5,]	0	0.0	0.0	0.0	0.0	0.9	0.0	0.3	0.0
[6,]	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5
[7,]	0	0.0	0.0	0.0	0.0	0.0	0.0	0.3	0.0
[8,]	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.6
[9,]	0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Vector of edge values:

```
edge_data_vector <- c(0.7, 0.1, 0.9, 0.1, 0.8, 0.2, 0.9, 0.4, 0.3,
0.3 ,0.5 ,0.6)
```

Matrix for the node positions:

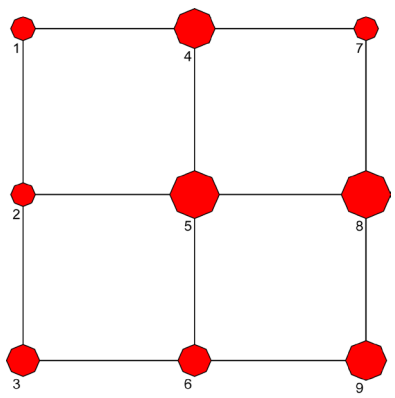
```
position_vector <- c(0,0,0,1,1,1,2,2,2, 2,1,0,2,1,0,2,1,0)

position_matrix <- matrix(position_vector, nrow=9)
```

	[,1]	[,2]
[1,]	0	2
[2,]	0	1
[3,]	0	0
[4,]	1	2
[5,]	1	1
[6,]	1	0
[7,]	2	2
[8,]	2	1
[9,]	2	0

Vector for the node values:

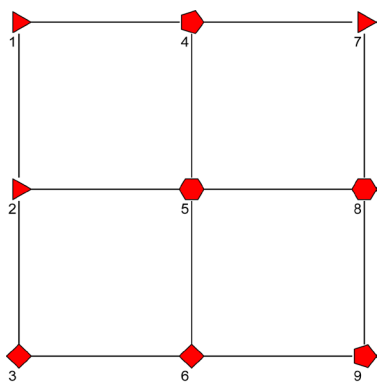
```
node_vector <- c(3,3,4,5,6,4,3,6,5)
```

Node-mark attributes



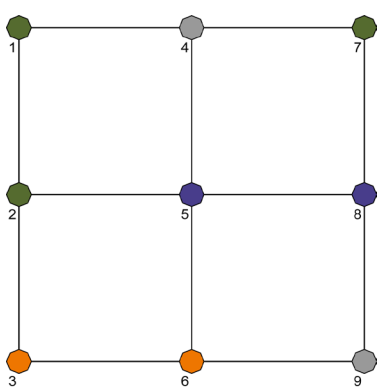
Setting size of each node.

```
gplot(data_matrix, coord=position_matrix, gmode="graph", jitter=FALSE,
displaylabels=TRUE, boxed.labels=FALSE, label.pos=1, vertex.cex=node_vector)
```



Setting number of sides for each node.

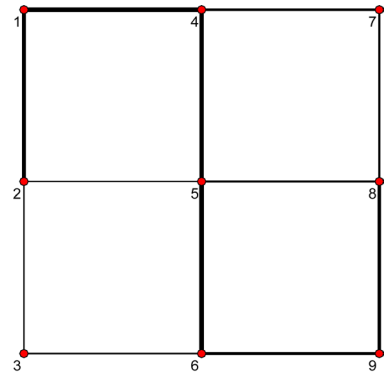
```
gplot(data_matrix, coord=position_matrix, gmode="graph", jitter=FALSE,
displaylabels=TRUE, boxed.labels=FALSE, label.pos=1, vertex.cex=3,
vertex.sides=node_vector)
```



Setting node color using the colors from the palette.

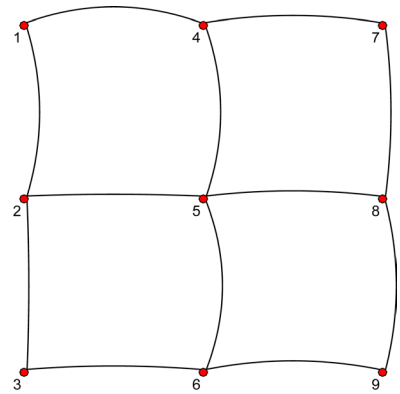
```
palette(c("black", "steelblue1", "darkolivegreen", "darkorange2",
"gray60", "darkslateblue"))

gplot(data_matrix, coord=position_matrix, gmode="graph", jitter=FALSE,
displaylabels=TRUE, boxed.labels=FALSE, label.pos=1, vertex.cex=3,
vertex.col=node_vector)

palette("default")
```

Edge-mark attributes



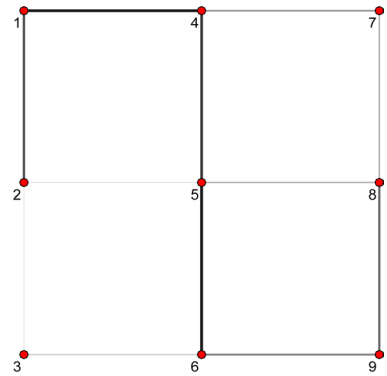
Edge width will be scaled by the data value in matrix if `edge.lwd` is set to any value but zero.

```
gplot(data_matrix, coord=position_matrix, gmode="graph", jitter=FALSE,
displaylabels=TRUE, boxed.labels=FALSE, label.pos=1, edge.lwd=10)
```



Edge curvature can be set for each edge.

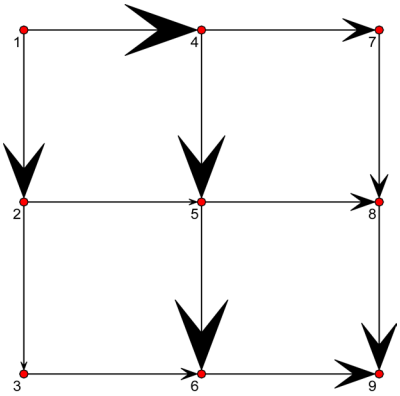
```
gplot(data_matrix, coord=position_matrix, gmode="graph", jitter=FALSE,
displaylabels=TRUE, boxed.labels=FALSE, label.pos=1, usecurve=TRUE,
edge.curve=data_matrix/10)
```



This example uses the `edge.col` function to set transparency of the edge. Transparency is set in the fourth position of the `rbg` function. Because the `rgb` function returns a vector, the output must be put in matrix form.

```
netSize <- dim(data_matrix)[1]
```

```
gplot(data_matrix, coord=position_matrix, gmode="graph", jitter=FALSE,
displaylabels=TRUE, boxed.labels=FALSE, label.pos=1, edge.lwd=5,
edge.col=matrix(rgb(0, 0, 0, data_matrix), nrow=netSize))
```



Size of arrowheads can be controlled with a vector of edge values (a matrix will not work).

```
gplot(data_matrix, coord=position_matrix, gmode="digraph",
jitter=FALSE, displaylabels=TRUE, boxed.labels=FALSE, label.pos=1,
arrowhead.cex=edge_data_vector*10)
```