```
data <- read.csv('dolphin_network.csv')
View(data)
```

```
library(igraph)
```

```
##
## Attaching package: 'igraph'

## The following objects are masked from 'package:stats':
##
##     decompose, spectrum

## The following object is masked from 'package:base':
##
##     union
```

```
help("read.graph")
graph_from_data_frame
```

```
## function (d, directed = TRUE, vertices = NULL)
## {
##     d <- as.data.frame(d)
##     if (!is.null(vertices)) {
##         vertices <- as.data.frame(vertices)
##     }
##     if (ncol(d) < 2) {
##         stop("the data frame should contain at least two columns")
##     }
##     if (any(is.na(d[, 1:2]))) {
##         warning("In 'd' 'NA' elements were replaced with string \"NA\"")
##         d[, 1:2][is.na(d[, 1:2])] <- "NA"
##     }
##     if (!is.null(vertices) && any(is.na(vertices[, 1]))) {
##         warning("In 'vertices[,1]' 'NA' elements were replaced with string \"NA\"")
##         vertices[, 1][is.na(vertices[, 1])] <- "NA"
##     }
##     names <- unique(c(as.character(d[, 1]), as.character(d[,
##         2])))
##     if (!is.null(vertices)) {
##         names2 <- names
##         vertices <- as.data.frame(vertices)
##         if (ncol(vertices) < 1) {
##             stop("Vertex data frame contains no rows")
##         }
##         names <- as.character(vertices[, 1])
##         if (any(duplicated(names))) {
##             stop("Duplicate vertex names")
##         }
##         if (any(!names2 %in% names)) {
##             stop("Some vertex names in edge list are not listed in vertex data frame")
##         }
##     }
```

```
##     g <- make_empty_graph(n = 0, directed = directed)
##     attrs <- list(name = names)
##     if (!is.null(vertices)) {
##         if (ncol(vertices) > 1) {
##             for (i in 2:ncol(vertices)) {
##                 newval <- vertices[, i]
##                 if (inherits(newval, "factor")) {
##                   newval <- as.character(newval)
##                 }
##                 attrs[[names(vertices)[i]]] <- newval
##             }
##         }
##     }
##     g <- add_vertices(g, length(names), attr = attrs)
##     from <- as.character(d[, 1])
##     to <- as.character(d[, 2])
##     edges <- rbind(match(from, names), match(to, names))
##     attrs <- list()
##     if (ncol(d) > 2) {
##         for (i in 3:ncol(d)) {
##             newval <- d[, i]
##             if (inherits(newval, "factor")) {
##                 newval <- as.character(newval)
##             }
##             attrs[[names(d)[i]]] <- newval
##         }
##     }
##     g <- add_edges(g, edges, attr = attrs)
##     g
## }
## <bytecode: 0x7fcce910f778>
## <environment: namespace:igraph>
```

```
g <- graph_from_data_frame(data)
```

```
# This graph in DN has 62 nodes and 158 edges, has labels
summary(g)
```

```
## IGRAPH c0f7945 DN-- 62 158 --
## + attr: name (v/c)
```

```
# Number of nodes
vcount(g)
```

```
## [1] 62
```

```
# Number of edges
ecount(g)
```

```
## [1] 158
```

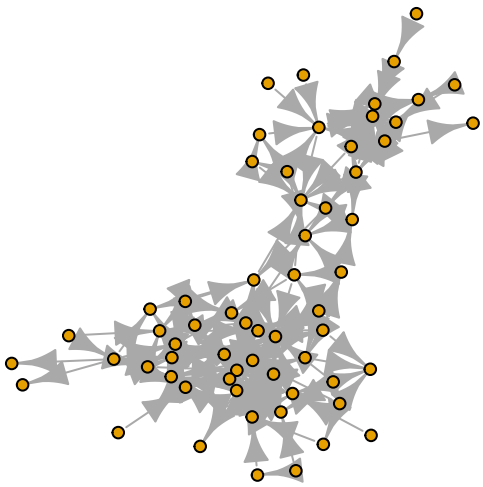
```
# Print nodes
V(g)
```

```
## + 62/62 vertices, named, from c0f7945:
##  [1] 15 16 41 43 48 18 20 27 28 29 37 42 55 11 45 62 9  60 52 10 14 57 58 31 21
## [26] 38 46 33 30 34 17 25 35 39 44 51 53 19 56 23 26 32 22 36 61 50 40 59 47 54
## [51] 1  2  3  4  5  6  7  8  12 13 24 49
```

```
# Print a sample of 10 edges
E(g)[sample(1:ecount(g), 10)]
```

```
## + 10/158 edges from c0f7945 (vertex names):
##  [1] 53->15 31->8  28->26 14->7  38->15 38->37 41->16 30->19 41->38 46->30
```

```
# Visualizing the graph
coords = layout.fruchterman.reingold(g)
plot(g, layout=coords, vertex.label=NA, vertex.size=5)
```



#g1 <- graph(edges=c(1, 2, 2, 3, 3, 4,4,5,5,1), directed=T) #diameter(g1) # gives 4 #g2 <- graph(edges=c(1, 2, 2, 3, 3, 4,4,5,5,1), directed=F) #diameter(g2) # gives 2 "'