

# Networks in R - Project 3 - R Package Dependencies

## Project Description

The network chosen for the project is the set of R packages built from two subsets

- the base packages
- the packages available on CRAN

with dependencies between them. The network data was gathered by using the `01_build_net.R` script.

There are four types of links between packages

- “Depends”. Directive attempts to ensure that a function from another package is available by attaching the other package to the main search path (i.e. the list of environments returned by `search()`). Which sometimes can lead to collisions with names. It’s not recommended type of dependency).
- “Imports”. Packages listed here must be present for your package to work. In fact, any time your package is installed, those packages will, if not already present, be installed on your computer (`devtools::load_all()` also checks that the packages are installed). However, it does not mean that it will be attached along with your package. The best practice is to explicitly refer to external functions using the syntax `package::function()`.
- “Suggests”. Your package can use these packages, but doesn’t require them. You might use suggested packages for example datasets, to run tests, build vignettes, or maybe there’s only one function that needs the package. Packages listed in Suggests are not automatically installed along with your package.
- “Enhances”. Packages listed here are “enhanced” by your package. Typically, this means you provide methods for classes defined in another package (a sort of reverse Suggests). But it’s hard to define what that means, so I don’t recommend using Enhances.

The “Suggests” and “Enhances” relations are out of scope for the project as they don’t impose hard dependency i.e. a package may work without them. Further, we consider only the “Depends” and “Imports” relations and consider them equal from our research’s point of view.

The motivation is to investigate into what are the most ‘powerful’ R packages in terms of their relations to the rest of packages.

## Data Format

The network data is saved in two formats:

- the GML format in the `r_packages.gml` file,
- the Pajek format in the `r_packages.net` file.

## Basic Network Characteristics

The nodes represent web pages, each page describes some Wikipedia social norm. The links are the HTTP reference from one page to another.

It’s a directed graph by its nature, no weights assigned to the links.

Property	Value
Vertices, number	12014
Arcs, number	44474

Property	Value
Average degree	7.4
Diameter	24
Acyclic?	FALSE

Interesting to note there are isolated packages i.e. without any relations to any other packages, their percentage is

```
## [1] 13.3
```

## Top Nodes

The most popular packages in terms of others depending on them, the

The first 50 nodes with the largest number of in-degree. The high number of total degree for a node is expectedly provided by incoming dependencies.

## [1] Page Name	Type	In-degree	Out-degree	Total Degree
## [1] -----				
## [1] stats	base	2812	7	2819
## [1] methods	base	2263	4	2267
## [1] utils	base	1721	0	1721
## [1] graphics	base	1503	0	1503
## [1] Rcpp	CRAN	1213	0	1213
## [1] MASS	CRAN	1144	6	1150
## [1] ggplot2	CRAN	1129	3	1132
## [1] grDevices	base	913	3	916
## [1] dplyr	CRAN	693	6	699
## [1] Matrix	CRAN	665	13	678
## [1] parallel	base	652	4	656
## [1] plyr	CRAN	551	1	552
## [1] mvtnorm	CRAN	481	20	501
## [1] stringr	CRAN	485	4	489
## [1] magrittr	CRAN	441	2	443
## [1] survival	CRAN	429	2	431
## [1] httr	CRAN	413	10	423
## [1] jsonlite	CRAN	417	5	422
## [1] sp	CRAN	404	2	406
## [1] lattice	CRAN	393	0	393
## [1] data.table	CRAN	379	8	387
## [1] foreach	CRAN	351	5	356
## [1] grid	base	351	1	352
## [1] reshape2	CRAN	346	4	350
## [1] igraph	CRAN	341	4	345
## [1] shiny	CRAN	279	2	281
## [1] tibble	CRAN	263	10	273
## [1] tidyr	CRAN	256	3	259
## [1] doParallel	CRAN	240	6	246
## [1] RColorBrewer	CRAN	242	3	245
## [1] coda	CRAN	226	3	229
## [1] raster	CRAN	229	0	229
## [1] XML	CRAN	219	2	221

```
## [1] zoo                CRAN                214            0            214
## [1] RCurl                CRAN                192            7            199
## [1] scales              CRAN                192            4            196
## [1] purrr               CRAN                193            1            194
## [1] nlme                 CRAN                176            6            182
## [1] lubridate           CRAN                178            2            180
## [1] gridExtra           CRAN                174            4            178
## [1] numDeriv            CRAN                172            3            175
## [1] xml2                 CRAN                169            3            172
## [1] ape                  CRAN                170            0            170
## [1] rgl                  CRAN                168            1            169
## [1] tools               base                166            0            166
## [1] splines             base                163            1            164
## [1] tcltk               base                158            4            162
## [1] boot                CRAN                161            1            162
## [1] digest              CRAN                156            0            156
## [1] mgcv                CRAN                153            2            155
```

The most dependent packages (sorted in the descending order of out-degree)

## [1] Page Name	Type	In-degree	Out-degree	Total Degree
## [1] -----				
## [1] ggnetwork	CRAN	3	40	43
## [1] dHSIC	CRAN	1	39	40
## [1] seriation	CRAN	11	37	48
## [1] Tcomp	CRAN	0	36	36
## [1] uqr	CRAN	0	32	32
## [1] dagR	CRAN	0	31	31
## [1] merror	CRAN	0	30	30
## [1] sisVIVE	CRAN	0	30	30
## [1] pumilioR	CRAN	0	29	29
## [1] qboxplot	CRAN	0	29	29

## Components

### Density

The proportion of present edges from all possible edges in the network.

```
## [1] 0.0003081535
```

The nodes along the first found path of the diameter distance.

```
V(net)$name[get.diameter(net)]
```

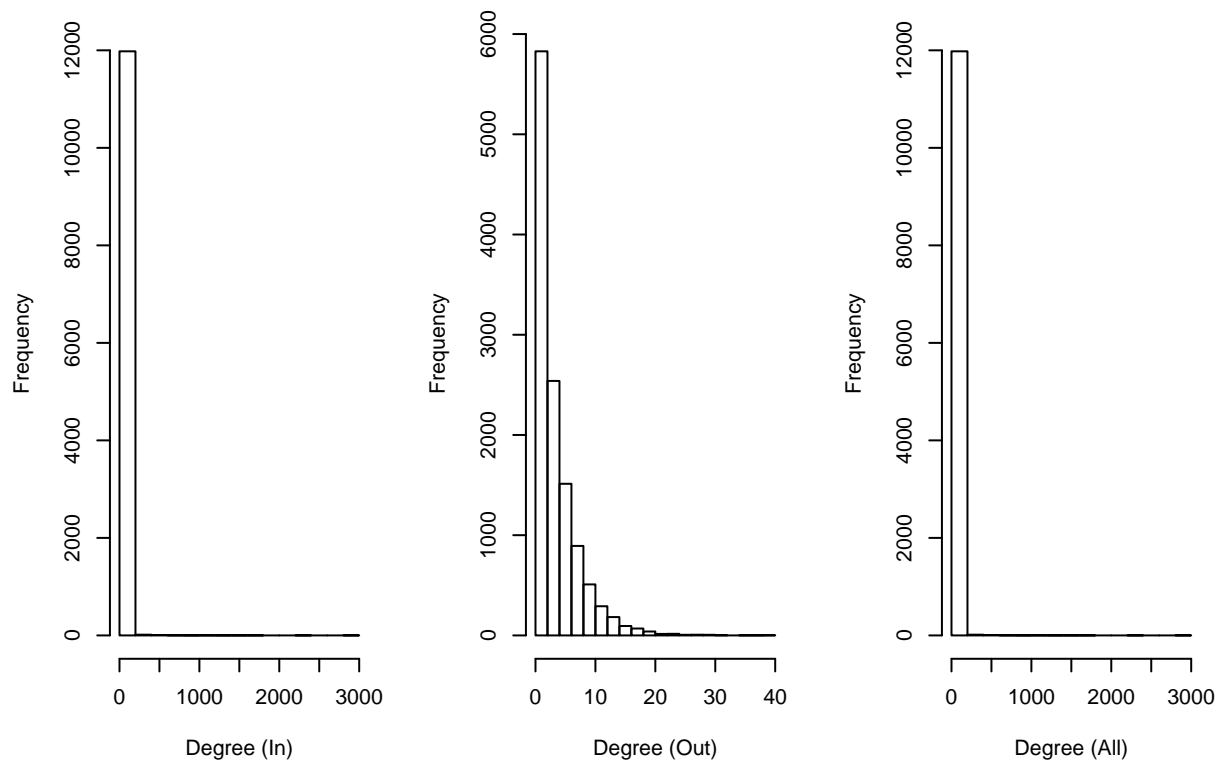
```
## [1] "adhoc"      "ade4"      "methods"   "corrplot"  "wavethresh"
## [6] "data.table" "reshape2"  "jsonlite"  "checkmate" "pls"
## [11] "httpuv"     "fastmatch" "kinship2"  "ddR"       "Rlibeemd"
## [16] "googleVis"  "sqldf"     "tcltk"     "mosaic"    "move"
## [21] "BiasedUrn"  "mitools"   "daewr"     "picante"   "uuid"
```

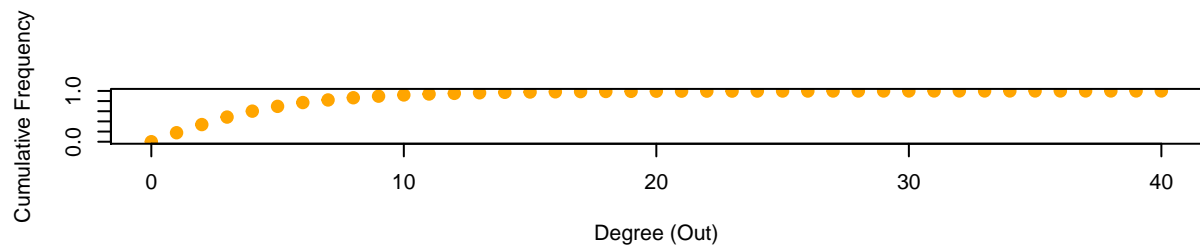
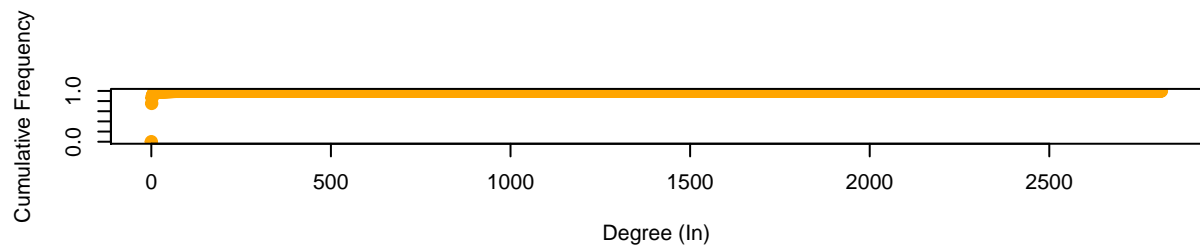
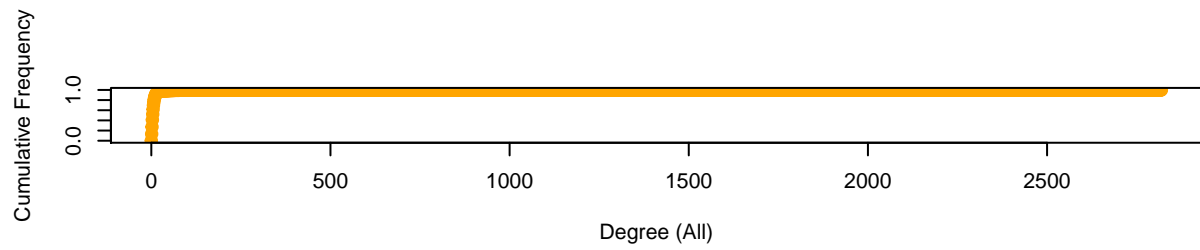
# Degree

The degree distribution (in-degree, out-degree, total)

```
# Degree
net_deg_in  <- degree(net, mode = "in")
net_deg_out <- degree(net, mode = "out")
net_deg_all <- degree(net, mode = "all")

par(mfrow = c(1,3))
hist(net_deg_in,  breaks = 20, freq = T, main = "", xlab = "Degree (In)")
hist(net_deg_out, breaks = 20, freq = T, main = "", xlab = "Degree (Out)")
hist(net_deg_all, breaks = 20, freq = T, main = "", xlab = "Degree (All)")
```

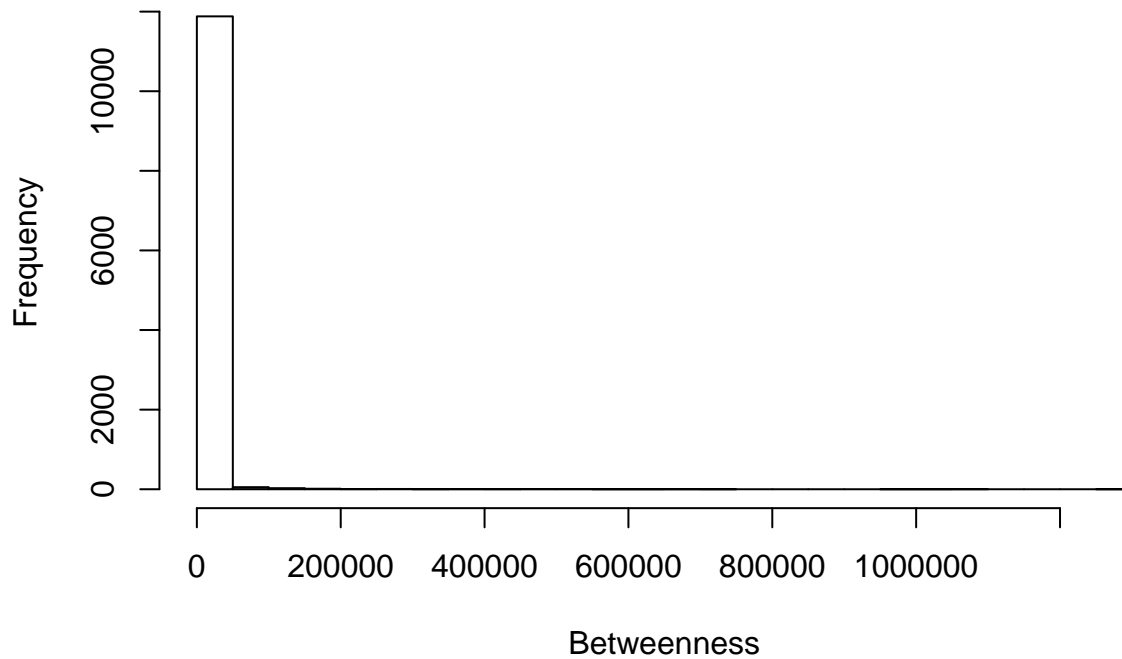




→

## Betweenness

```
net_bt = betweenness(net, v = V(net), directed = TRUE)
hist(net_bt, breaks = 30, freq = T, main = "", xlab = "Betweenness")
```



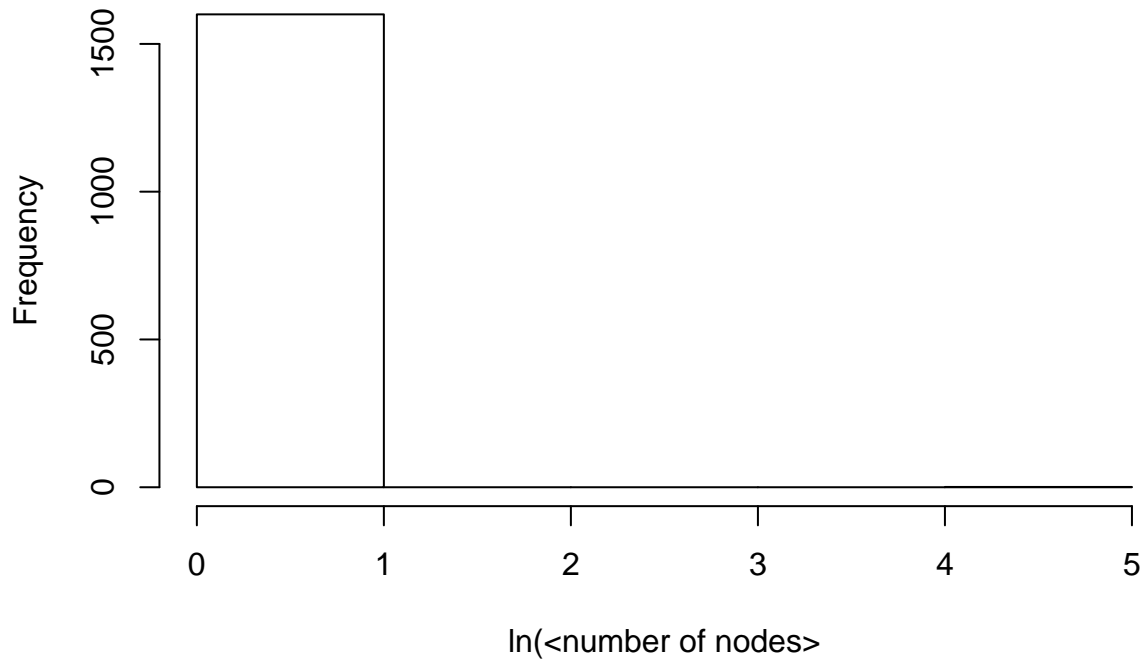
```
# net_eb <- cluster_edge_betweenness(net, directed = TRUE)
# plot_dendrogram(net_eb)
```

## Subnetworks

In trying to determine subnetworks with the we find there are hundrends of one node networks (isolated nodes) and one large subnetwork of 10406 nodes.

That's shown on the histogram below where x is ln of the size of each of those networks

**Histogram of  $\log(\text{net\_sn\_len}, 10)$**

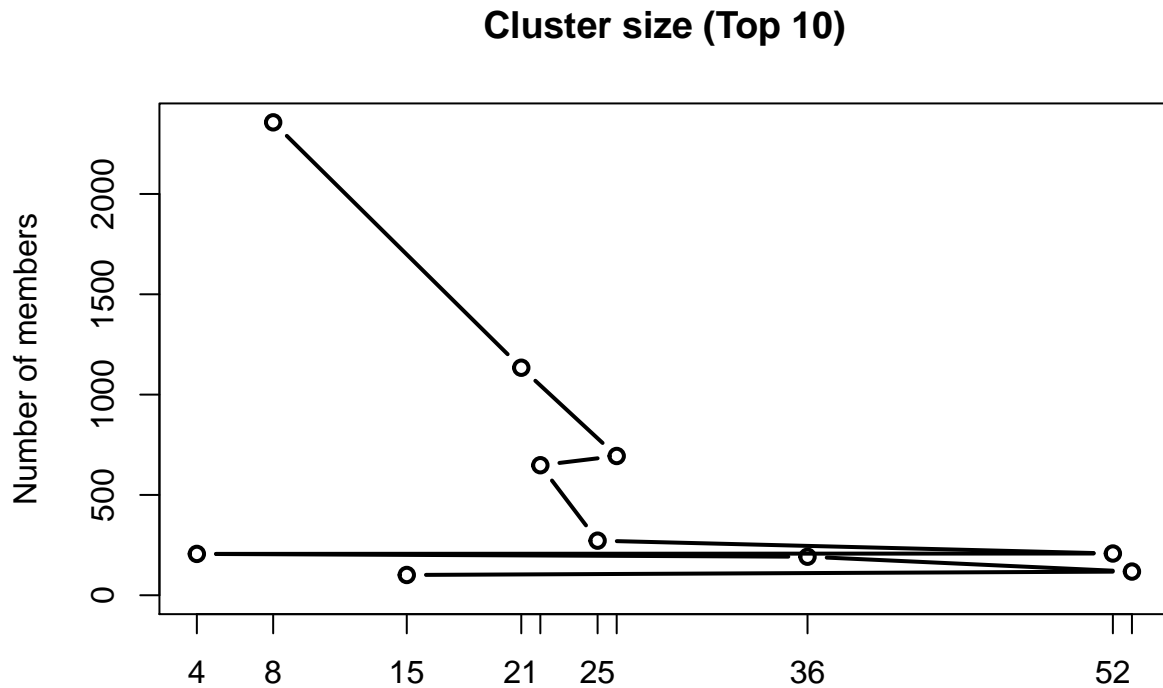


## Communities

TBC...

Determine communities using walktrap algorithm (igraph)

```
##
##      8    21    26    22    25    52     4    36    53    15
## 2357 1134  694  648  272  208  206  192  118  101
```



## Visualization

Visualization was made in Gephi with manual configuration. An effort of doing the same with igraph provides suboptimal results of nodes and edges overlapping each other.

The size of nodes reflects the number of in-degrees for that node. The color of node reflects which type ('base', 'CRAN') it belongs to.

The visualization demonstrates the network is centered around several large nodes.

image:

The visualization also saved as R\_packages\_vis.pdf.

## Ideas to Improve Report

1. Dendrogram? Will it work for such a large network?
2. Try the hierarchical clustering.
3. Provide more interpretation of the network.

## References

1. CRAN: <https://cran.r-project.org>



## Appendix A Technical Details of Report

This version of the report was built with:

```
devtools::session_info()
```

```
## Session info -----
##   setting  value
##   version  R version 3.4.3 (2017-11-30)
##   system   x86_64, mingw32
##   ui       RTerm
##   language en
##   collate   Russian_Russia.1251
##   tz        Europe/Moscow
##   date      2018-01-08

## Packages -----
##   package * version date          source
##   backports 1.1.2 2017-12-13 CRAN (R 3.4.3)
##   base      * 3.4.3 2017-12-06 local
##   compiler  3.4.3 2017-12-06 local
##   datasets  * 3.4.3 2017-12-06 local
##   devtools  1.13.4 2017-11-09 CRAN (R 3.4.3)
##   digest     0.6.13 2017-12-14 CRAN (R 3.4.3)
##   evaluate   0.10.1 2017-06-24 CRAN (R 3.4.3)
##   graphics  * 3.4.3 2017-12-06 local
##   grDevices * 3.4.3 2017-12-06 local
##   htmltools  0.3.6 2017-04-28 CRAN (R 3.4.3)
##   igraph     * 1.1.2 2017-07-21 CRAN (R 3.4.3)
##   knitr      1.18   2017-12-27 CRAN (R 3.4.3)
##   magrittr   1.5    2014-11-22 CRAN (R 3.4.3)
##   memoise    1.1.0  2017-04-21 CRAN (R 3.4.3)
##   methods   * 3.4.3 2017-12-06 local
##   pkgconfig  2.0.1  2017-03-21 CRAN (R 3.4.3)
##   Rcpp       0.12.14 2017-11-23 CRAN (R 3.4.3)
##   rmarkdown  1.8     2017-11-17 CRAN (R 3.4.3)
##   rprojroot  1.3-1   2017-12-18 CRAN (R 3.4.3)
##   stats     * 3.4.3 2017-12-06 local
##   stringi    1.1.6  2017-11-17 CRAN (R 3.4.2)
##   stringr    1.2.0  2017-02-18 CRAN (R 3.4.3)
##   tools      3.4.3  2017-12-06 local
##   utils     * 3.4.3 2017-12-06 local
##   withr      2.1.1  2017-12-19 CRAN (R 3.4.3)
##   yaml       2.1.16 2017-12-12 CRAN (R 3.4.3)
```