Introduction to Social Network Analysis with igraph

Robert Ackland & Timothy Graham

13 July 2015

In this session we conduct basic social network analysis using *igraph*

Setting up

First load the *igraph* package

```
library(igraph)
```

```
##
## Attaching package: 'igraph'
##
## The following objects are masked from 'package:stats':
##
##     decompose, spectrum
##
## The following object is masked from 'package:base':
##
##     union
```

Now read in the graphml file. This YouTube comment network was collected using *SociaMediaLab* using the following 5 videos on "Fracking Australia":

https://www.youtube.com/watch?v=XrE7LzZCn1E

https://www.youtube.com/watch?v=PELxZ3K2o0c

https://www.youtube.com/watch?v=Xfo0hpVrtrs

https://www.youtube.com/watch?v=7Toi2s05Exg

https://www.youtube.com/watch?v=BBSVLGf7zPI

```
#g <- read.graph("http://voson.anu.edu.au/papers/CASSBigDataAnalysis2015/Sat_Jun_27_13-04-34_2015_EST_Y
g <- read.graph("Sat_Jun_27_13-04-34_2015_EST_YoutubeActorNetwork_FiveFracking.graphml", format="graphml
```

Some basic information on the network

```
g   #Information on the igraph graph object
```

```
## IGRAPH DNW- 137 359 --
## + attr: name (v/c), label (v/c), id (v/c), weight (e/n)
## + edges (vertex names):
##  [1] Jaya Chela Drolma->Jaya Chela Drolma
##  [2] Jaya Chela Drolma->Jaya Chela Drolma
##  [3] Lenard Bendell   ->Michael John Battista
```

```
##  [4] Francine Hirst   ->Lady Fran W
##  [5] Lady Fran W      ->Francine Hirst
##  [6] Francine Hirst   ->Betty Tuohey
##  [7] Betty Tuohey     ->Francine Hirst
##  [8] Francine Hirst   ->Rea de Miranda
## + ... omitted several edges
```

```
V(g)               #list of vertices
```

```
## + 137/137 vertices, named:
##   [1] Jaya Chela Drolma
##   [2] Lenard Bendell
##   [3] Francine Hirst
##   [4] Lady Fran W
##   [5] Betty Tuohey
##   [6] Rea de Miranda
##   [7] Jon O'Hare
##   [8] DIY Legal Kits
##   [9] Adriano Devon
##  [10] Slayermetallica69
## + ... omitted several vertices
```

```
E(g)               #list of edges
```

```
## + 359/359 edges (vertex names):
##  [1] Jaya Chela Drolma->Jaya Chela Drolma
##  [2] Jaya Chela Drolma->Jaya Chela Drolma
##  [3] Lenard Bendell   ->Michael John Battista
##  [4] Francine Hirst   ->Lady Fran W
##  [5] Lady Fran W      ->Francine Hirst
##  [6] Francine Hirst   ->Betty Tuohey
##  [7] Betty Tuohey     ->Francine Hirst
##  [8] Francine Hirst   ->Rea de Miranda
##  [9] Rea de Miranda   ->Francine Hirst
## [10] Jon O'Hare       ->DIY Legal Kits
## + ... omitted several edges
```

```
V(g)[36]           #accessing particular vertex
```

```
## + 1/137 vertex, named:
## [1] LEO SAYER
```

```
E(g)[1]            #accessing particular edge
```

```
## + 1/359 edge (vertex names):
## [1] Jaya Chela Drolma->Jaya Chela Drolma
```

```
V(g)$id            #list of "id" (vertex) attributes
```

```
##   [1] "n0"   "n1"   "n2"   "n3"   "n4"   "n5"   "n6"   "n7"   "n8"   "n9"
##  [11] "n10"  "n11"  "n12"  "n13"  "n14"  "n15"  "n16"  "n17"  "n18"  "n19"
##  [21] "n20"  "n21"  "n22"  "n23"  "n24"  "n25"  "n26"  "n27"  "n28"  "n29"
##  [31] "n30"  "n31"  "n32"  "n33"  "n34"  "n35"  "n36"  "n37"  "n38"  "n39"
##  [41] "n40"  "n41"  "n42"  "n43"  "n44"  "n45"  "n46"  "n47"  "n48"  "n49"
##  [51] "n50"  "n51"  "n52"  "n53"  "n54"  "n55"  "n56"  "n57"  "n58"  "n59"
##  [61] "n60"  "n61"  "n62"  "n63"  "n64"  "n65"  "n66"  "n67"  "n68"  "n69"
##  [71] "n70"  "n71"  "n72"  "n73"  "n74"  "n75"  "n76"  "n77"  "n78"  "n79"
##  [81] "n80"  "n81"  "n82"  "n83"  "n84"  "n85"  "n86"  "n87"  "n88"  "n89"
##  [91] "n90"  "n91"  "n92"  "n93"  "n94"  "n95"  "n96"  "n97"  "n98"  "n99"
## [101] "n100" "n101" "n102" "n103" "n104" "n105" "n106" "n107" "n108" "n109"
## [111] "n110" "n111" "n112" "n113" "n114" "n115" "n116" "n117" "n118" "n119"
## [121] "n120" "n121" "n122" "n123" "n124" "n125" "n126" "n127" "n128" "n129"
## [131] "n130" "n131" "n132" "n133" "n134" "n135" "n136"
```

```
E(g)$weight        #list of "weight" (edge) attributes
```

```
##   [1]  4  4  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2
##  [24]  2  2  4  4  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2
##  [47]  2  2  2  2  1  1 14 12 14 12 14 12 14 12 14 12 14  2  2 12 14  2  1
##  [70]  1  2  2  2  2  2  2  2 14 12 14 12 14 12 14 12 14 12 14  2  2 12 14
##  [93]  2  2  2  2  2  2  2  2  1  8  2  2  8  2  2  2  2  6  6  6  2  2  8
## [116]  2  2  2  8  2  1  1  1  1  1  1  3  3  3  3  3  3  1  1  1  1  1  1
## [139]  1  1  4  1  9 11  9 11 11 11  9  2  2  2 13  4  1 13 13  4  1  1 13
## [162]  4 13  1  1  3  1  1  3  3 13 13  4 13  1  9  1  1  1  1  1  1  1 11
## [185]  9  2  2  3  3  3 11 11  9  9 11  4 13  4  9 11 13 13 13  4 11  9  4
## [208] 11  1  2  2 13  1  3  3  2  3  2  1  1  2  4  4  1  4  4  4  4  4  3
## [231]  1  4  3  1  4  3  4  4  4  4  3  4  3  4  3  1  1  1  1  2  2  2  2
## [254]  1  1  1  2  1  2  1  1  1  1  2  2  1  1  1  1  1  1  1  1  1  1  2
## [277]  1  1  1  2  1  1  1  1  3  1  1  1  2  3  3  2  1  1  1  1  8  2  2
## [300]  8  2  2  2  2  6  6  6  2  2  8  2  2  2  8  2  4  4  4  4  2  4  2
## [323]  4  2  4  4  4  4  2  4  2  4  2  6  6  6  6  6  6  2  2  6  2  6  2
## [346]  6  6  6  6  6  6  6  2  2  6  2  6  2  6
```

```
vcount(g)          #number of nodes in network
```

```
## [1] 137
```

```
length(V(g))       #another way
```

```
## [1] 137
```

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

```
## [1] 359
```

```
length(E(g))       #another way
```

```
## [1] 359
```

```
list.vertex.attributes(g)     #list of the vertex attributes
```

```
## [1] "name"  "label" "id"
```

```
list.edge.attributes(g)       #list of the edge attributes
```

```
## [1] "weight"
```

Network maps

```
#png("csg_plot1.png", width=800, height=700)
plot(g, vertex.shape="none",edge.width=1.5, edge.curved = .5, edge.arrow.size=0.5, asp=9/16,margin=-0.15
```


```
#dev.off()
```

We notice that there are loops in the network, let's investigate this further...

```
is.simple(g)                  #test whether "simple" graph (no loops or multiple edges)
```

```
## [1] FALSE
```

```
g2 <- simplify(g)             #construct "simple" version of network
g2
```

```
## IGRAPH DNW- 137 142 --
## + attr: name (v/c), label (v/c), id (v/c), weight (e/n)
## + edges (vertex names):
##  [1] Lenard Bendell   ->Michael John Battista
##  [2] Francine Hirst   ->Lady Fran W
##  [3] Francine Hirst   ->Betty Tuohey
```

```
##  [4] Francine Hirst   ->Rea de Miranda
##  [5] Lady Fran W      ->Francine Hirst
##  [6] Betty Tuohey     ->Francine Hirst
##  [7] Rea de Miranda   ->Francine Hirst
##  [8] Jon O'Hare       ->DIY Legal Kits
## + ... omitted several edges
```

```
is.simple(g2)
```

```
## [1] TRUE
```

```
#png("csg_plot2.png", width=800, height=700)
plot(g2, vertex.shape="none",edge.width=1.5, edge.curved = .5, edge.arrow.size=0.5, asp=9/16,margin=-0.
```


```
#dev.off()
```

Looking at the connectivity of the graph

```
neighbors(g2,36)                    #Leo Sayer's network neighbors
```

```
## + 34/137 vertices, named:
##  [1] Ronald Martens   Donna Philben    rebecca hooper   Liniute ilegale
##  [5] seer charlotte   Operation Pravda Mike Bennett     James Doogue
##  [9] Rod Molyneux     David Anderson   Scott Miller     goodonya
## [13] Mark A           Christian Rioux  dfissell         Will Franklin
## [17] Margaret Henry   John King        Therese Mackay   TheMalcolmPowder
## [21] William Kenny    Squirrlyburt     ronleon62        Rod Ferguson
## [25] Jacqui Pyke      ozripper1        MrFillbee        Christine Dixon
## [29] Isabelle1956     Martha Gillespie Mark Cole        Don Bryant
## [33] NEV E            Jasmine Ham
```

```r
is.connected(g2, mode="weak")        #this is not a weakly connected component
```

```
## [1] FALSE
```

```r
cc <- clusters(g2)              #information on connected components
head(cc$membership)                   #which component each node is assigned to
```

```
## Jaya Chela Drolma    Lenard Bendell    Francine Hirst       Lady Fran W
##                 1                 2                 3                 3
##      Betty Tuohey    Rea de Miranda
##                 3                 3
```

```r
cc$csize                        #size of each component
```

```
##  [1]  1  2  4  3  5  4  1  3  3  4  2  2  2  2  2  3 66  2  1  1  2  3  2
## [24]  3  2  2  3  2  5
```

```r
cc$no                           #number of components
```

```
## [1] 29
```

```r
g3 <- induced_subgraph(g2, which(cc$membership == which.max(cc$csize)))    #subnetwork - giant componen

#png("csg_plot3.png", width=800, height=700)
plot(g3, vertex.shape="none",edge.width=1.5, edge.curved = .5, edge.arrow.size=0.5, asp=9/16,margin=-0.
```


```r
#dev.off()
```

Node centrality

```r
head(degree(g3, mode="in"))                    #node indegree
```

```
##     Ronald Martens          LEO SAYER          Pearly403
##                  2                 22                  0
##      Donna Philben       Suzi Dhnaram My Own Kind Of Man
##                  1                  0                  0
```

```r
head(degree(g3, mode="out"))                   #node outdegree
```

```
##     Ronald Martens          LEO SAYER          Pearly403
##                  3                 34                  1
##      Donna Philben       Suzi Dhnaram My Own Kind Of Man
##                  1                  1                  1
```

```r
ind <- strength(g3, mode="in")             #node indegree, using edge weights
V(g3)[order(ind, decreasing=T)[1:3]]       #top-5 nodes, based on (weighted) indegree
```

```
## + 3/66 vertices, named:
## [1] LEO SAYER    Rod Molyneux James Doogue
```

```r
head(closeness(g3))                        #closeness centrality
```

```
##     Ronald Martens          LEO SAYER          Pearly403
##       0.0005521811       0.0005740528       0.0005216484
##      Donna Philben       Suzi Dhnaram My Own Kind Of Man
##       0.0005617978       0.0005411255       0.0005411255
```

```r
head(betweenness(g3))                      #betweenness centrality
```

```
##     Ronald Martens          LEO SAYER          Pearly403
##                 74               1331                  0
##      Donna Philben       Suzi Dhnaram My Own Kind Of Man
##                  0                  0                  0
```

```r
head(evcent(g3)$vector)                    #eigenvector centrality
```

```
##     Ronald Martens          LEO SAYER          Pearly403
##        0.238868332        1.000000000        0.003354127
##      Donna Philben       Suzi Dhnaram My Own Kind Of Man
##        0.017552175        0.014041740        0.014041740
```

Network cohesion

```r
graph.density(g3)          #density
```

```
## [1] 0.02121212
```

```
transitivity(g3)            #(global) clustering coefficient - rel. frequency connected triples close t
```

```
## [1] 0.02616279
```

```
reciprocity(g3, mode="default")     #number of dyads with reciprocated (mutual) edges/number of dyads w
```

```
## [1] 0.3516484
```

```
reciprocity(g3, mode="ratio")       #total number of reciprocated edges/total number of edges
```

```
## [1] 0.2133333
```

Community identification

```
wt <- walktrap.community(g3)       #community identification (walktrap algorithm)
wt
```

```
## IGRAPH clustering walktrap, groups: 9, mod: 0.22
## + groups:
##   $`1`
##   [1] "Sally Bowman"  "Damh the Bard"
##
##   $`2`
##   [1] "Christian Rioux" "Margaret Henry"  "chris cole"
##   [4] "ruth WILLIAMS"   "drufimusic"
##
##   $`3`
##   [1] "Michelle Raadschelders" "Pavel Britov"
##   [3] "Kenneth Burchett"
##   + ... omitted several groups/vertices
```

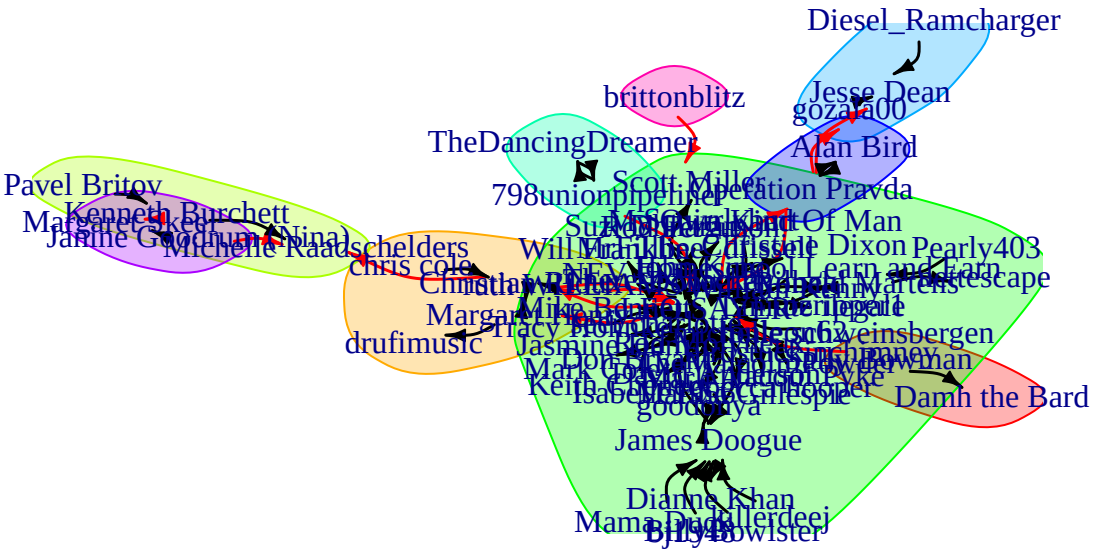
```
length(wt)                         #number of communities
```

```
## [1] 9
```

```
sizes(wt)                          #community sizes
```

```
## Community sizes
##  1  2  3  4  5  6  7  8  9
##  2  5  3 46  2  3  2  2  1
```

```
#png("csg_plot4.png", width=800, height=700)
plot(wt, g3, vertex.shape="none",edge.width=1.5, edge.curved = .5, edge.arrow.size=0.5, asp=9/16,margin=
```


Diesel_Ramcharger
brittonblitz
Jesse Dean
gozaa400
TheDancingDreamer
Alan Bird
Pavel Britov
Kenneth Burchett
Janine Gooch
Michelle Raadschelders
chris cole
798unionpipeliner
Operation Pravda
Pearly403
drufimusic
Damh the Bard
James Doogue

```
#dev.off()
```