Networks: Project 2

Dmitry Donetskov

Contents

1 Problem Statement

Select an undirected labeled network with at least 50 nodes and average degree at least 4. Determine the graphlet spectra for its nodes. On their basis construct a dissimilarity between nodes and use it to cluster the nodes.

2 Network

The data is the network of American football games between Division IA colleges during regular season Fall 2000, as compiled by M. Girvan and M. Newman (Girvan and Newman (2002)). To quote

> The network we look at is a representation of the schedule of Division I games for the 2000 season: vertices in the graph represent teams (identified by their college names) and edges represent regular-season games between the two teams they connect. What makes this network interesting is that it incorporates a known community structure. The teams are divided into conferences containing around 8–12 teams each. Games are more frequent between members of the same conference than between members of different conferences, with teams playing an average of about seven intraconference games and four interconference games in the 2000 season.

Let's check if the clustering based on the graphlets add more information.

2.1 Basic Network Characteristics

Property	Value
Vertices	115
Edges	613
Directed	No

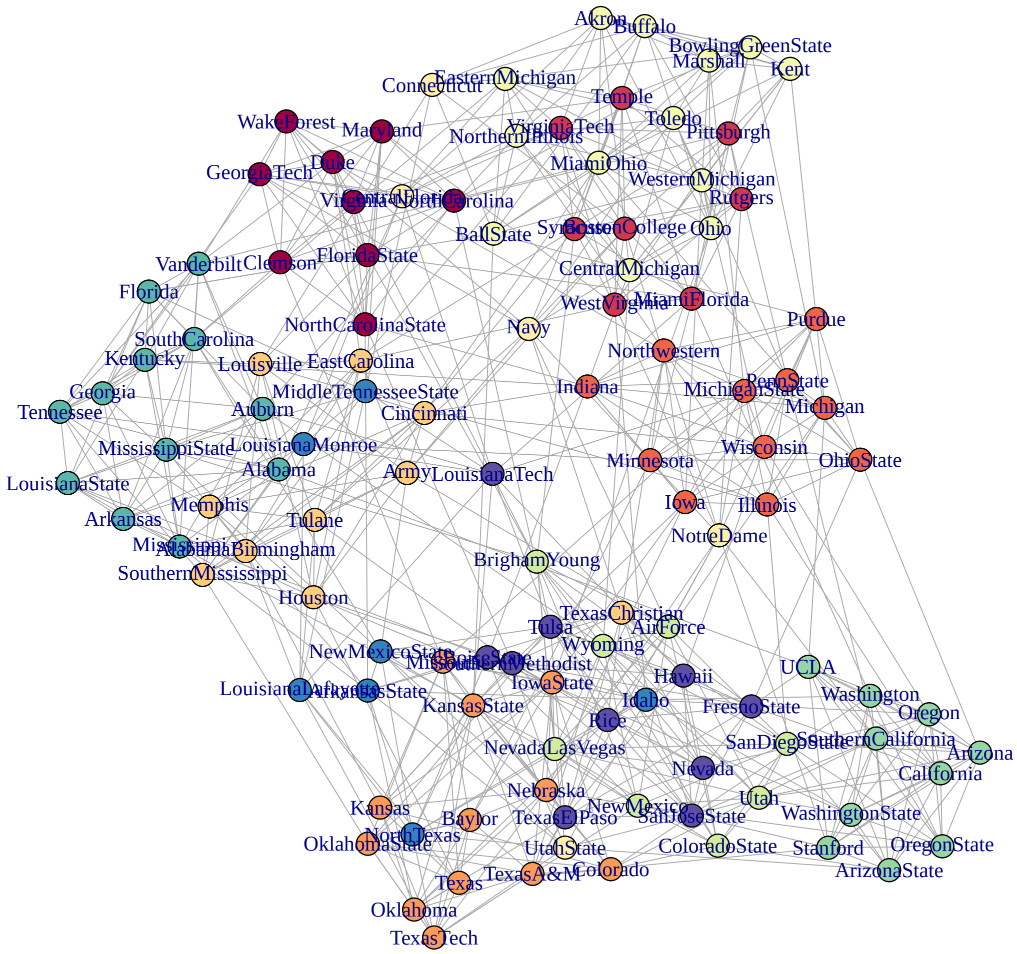
Property	Value
Weighted	No
Average degree	10.7
Diameter	4
Acyclic	No
Edge density	0.0935163996948894
Average Path Length	2.50816170861937
Transitivity (global)	0.407239819004525

The network coloured according to divisions each club belongs to.

```
pal = colorRampPalette(brewer.pal(11,"Spectral"))(length(unique(V(net)$value)))
node_colors  <- pal[V(net)$value + 1]
V(net)$color <- node_colors

coords_fr = layout.fruchterman.reingold(net, weights=E(net)$weight)

# igraph plot options
igraph.options(vertex.size=8, edge.width=0.75)
plot(net, layout=coords_fr, vertex.size = 5, vertex.color=V(net)$color)
```



3 Analysis

The orca package is used to find associations of nodes with the 4-node orbits.

3.1 Getting Orbits

```
# convert to the 'edgelist' format required by orca
net_edgelist <- as.data.frame(as_edgelist(net))
net_edgelist <- transform(net_edgelist, V1 = as.integer(net_edgelist$V1), V2 = as.integer(net_edgelist$V
```

Printing the graphlet spectra.

```r
# convert to the 'edgelist' format required by orca
net_orbits        <- count4(net_edgelist)
net_orbits_scaled <- scale(net_orbits)      # scale the data

print('Found orbits for the nodes are')
```

```
## [1] "Found orbits for the nodes are"
```

```r
print(net_orbits)
```

```
##        O0 O1 O2 O3  O4  O5  O6  O7 O8  O9 O10 O11 O12 O13 O14
##   [1,] 12 71 43 23 433 575 116  66 11 105 158 113  13   6  35
##   [2,] 12 58 37 29 363 461  98  49 13  81 147 108   2   7  56
##   [3,] 12 72 41 25 446 474 145  47 26  52 178 117  44  35  21
##   [4,] 12 73 44 22 424 527 145  73 32  68 152  91  30  39  17
##   [5,] 11 62 32 23 394 389 111  34 21  70 167  90  11   6  35
##   [6,] 12 64 42 24 371 417 100  59 34  58 159  99  30  45  17
##   [7,] 12 77 45 21 472 588 162  80 28  79 141  90  31  30  20
##   [8,] 12 68 40 26 391 496 131  61 28  81 136  90  30  37  32
##   [9,] 11 56 29 26 306 275  87  26 33  46 139  76  40  31  32
##  [10,] 11 57 29 26 350 327  76  24 17  75 175  83  17  23  35
##  [11,] 10 58 25 20 343 265 109  23 29  47 131  49  34  33  15
##  [12,] 10 64 31 14 412 393 129  38 20  75 119  62  12  10  10
##  [13,] 10 63 30 15 367 360 120  37 27  71 119  56  17  17  10
##  [14,] 11 69 34 21 412 445 125  46 18  82 143  69  33  30  20
##  [15,] 10 61 28 17 358 308 118  24 32  61 123  68  27  16  12
##  [16,] 12 74 43 23 422 547 123  70 24  99 157  90  31  40  20
##  [17,] 11 65 32 23 418 452 117  40  9  92 170  79   5  10  36
##  [18,] 11 67 36 19 403 434 135  52 27  60 124  70  29  28  15
##  [19,] 11 55 33 22 335 296  84  34 32  43 156  85  17  25  21
##  [20,] 11 56 31 24 341 347  72  31 16  73 161  88  10  10  36
##  [21,] 11 64 35 20 366 392 110  47 27  63 134  71  30  32  15
##  [22,] 11 60 30 25 350 368  96  34 17  76 140  69  29  30  32
##  [23,] 11 54 27 28 308 281  81  23 20  54 143  64  38  46  32
##  [24,] 11 58 30 25 336 309  82  24 30  68 173  92  19  14  35
##  [25,] 10 73 35 10 437 533 144  60 16 124  86  50   3   0  10
##  [26,] 11 52 27 28 371 360  94  25  3  63 148  84   3   0  56
##  [27,] 10 63 30 15 375 366 124  38 26  74 121  54  14  18  10
##  [28,] 11 65 35 20 358 397 120  47 32  66 131  71  29  32  15
##  [29,]  9 59 24 12 363 332 100  28 13  92 108  38   3   8  10
##  [30,] 11 50 28 27 394 286  88  23 10  26 176  79   9  25  38
##  [31,] 11 54 29 26 360 345  76  31  7  66 167  71   7  26  37
##  [32,] 11 63 34 21 332 365  98  37 35  77 153  87  21  21  20
##  [33,] 11 70 34 21 411 440 133  46 26  84 144  69  32  30  20
##  [34,] 10 42 16 29 291 197  63   7  7  47 163  50   2   7  56
##  [35,] 11 58 34 21 332 329  85  37 32  58 141  87  23  21  20
##  [36,] 11 61 31 24 365 405 108  37 17  84 155  76   8  16  36
##  [37,]  8 67 24  4 365 381 141  34 13 113  46  20   8   2   0
##  [38,] 11 46 26 29 362 291  79  23  6  34 153  79   2   7  56
##  [39,] 11 67 38 17 372 444 111  55 30  84 122  79  22  19  12
##  [40,] 11 65 32 23 429 370 132  33 23  45 166  77  34  35  20
##  [41,] 11 70 36 19 376 442 128  52 37  93 126  71  29  26  16
##  [42,] 10 54 21 24 346 273  76  16 10  78 183  52   5  16  36
##  [43,]  7 60 18  3 322 276 139  22 18  91  22  11  13   2   0
```

```
##  [44,] 11 66 39 16 383 436 115  62 35  77 124  72   15   21   10
##  [45,] 11 67 35 20 390 424 121  37 19  70 138  93   35   18   17
##  [46,] 11 50 26 29 349 329  85  23  7  61 153  79    2    7   56
##  [47,] 11 55 26 29 377 319  86  16 12  64 185  93   11    0   56
##  [48,] 11 71 34 21 408 444 135  46 26  86 150  69   32   30   20
##  [49,] 11 68 35 20 394 452 118  47 20  90 146  76   21   22   20
##  [50,] 11 53 25 30 371 314  82  21  9  63 188  74    5   14   56
##  [51,]  9 56 22 14 368 290  93  14  5  73 114  53   17    6   11
##  [52,] 11 54 29 26 330 312  80  31 18  55 132  67   30   34   33
##  [53,] 10 63 26 19 365 318 127  26 25  62 137  52   31   26   16
##  [54,] 12 61 36 30 408 455  95  40  7  78 182 116   10    8   56
##  [55,] 10 53 23 22 303 239  74  16 24  64 166  55   12   26   23
##  [56,] 11 62 32 23 371 411 107  35 16  82 143  88   17    7   35
##  [57,] 10 59 26 19 350 296 106  26 24  57 127  51   30   28   15
##  [58,] 10 60 28 17 390 343 111  34 16  61 127  52   19   18   16
##  [59,] 10 81 40  5 491 646 183  84 12 135  43  33    8    2    1
##  [60,]  8 58 22  6 377 303 116  25 12  76  67  27    6    3    1
##  [61,] 11 66 34 21 413 400 122  46 23  57 150  69   32   30   20
##  [62,] 11 70 37 18 399 477 143  55 30  81 125  78   19   12   20
##  [63,] 11 67 35 20 358 418 115  43 25  78 128  79   37   28   15
##  [64,]  9 68 31  5 426 454 142  53 14 100  52  28    5    2    1
##  [65,] 11 68 34 21 423 428 129  46 21  69 150  69   30   30   20
##  [66,] 11 66 35 20 365 397 133  47 37  62 133  71   28   32   15
##  [67,] 11 66 36 19 398 419 126  50 28  61 143  75   21   24   16
##  [68,] 12 58 34 32 403 395  89  23  6  61 193 131   19    9   57
##  [69,] 11 58 29 26 341 352  98  31 15  70 138  67   30   34   33
##  [70,] 11 76 42 13 456 602 139  72 13 122 104  79   14    4   10
##  [71,] 11 66 35 20 357 419 120  48 26  72 127  69   31   33   15
##  [72,] 10 57 25 20 342 299 102  24 22  64 144  55   13   18   23
##  [73,] 11 66 34 21 380 402 114  44 30  83 146  70   26   34   17
##  [74,] 11 56 26 29 384 330  96  16 11  63 184  93   12    0   56
##  [75,] 11 68 34 21 380 402 130  45 37  78 148  68   28   35   17
##  [76,] 10 59 26 19 372 312 105  24 15  57 143  56   23   24   16
##  [77,] 11 56 32 23 321 281  80  27 30  43 151  84   34   39   15
##  [78,] 11 58 29 26 358 350  98  32 13  61 144  64   30   37   32
##  [79,] 11 48 26 29 292 228  58  19 17  38 152  64   33   49   33
##  [80,] 11 56 29 26 350 341  91  26 13  62 167  80   13   23   36
##  [81,] 11 85 45 10 428 698 127  90  6 167  82  62   29   11    2
##  [82,] 11 70 35 20 395 434 140  48 32  77 136  69   33   33   15
##  [83,] 11 93 47  8 488 773 189 101 18 167  59  58   31    4    2
##  [84,] 11 51 24 31 373 280  82  13  8  46 198  81    8   15   56
##  [85,] 11 66 34 21 352 372 111  40 37  73 144  78   35   30   17
##  [86,]  9 61 23 13 381 301 137  23 24  63 107  41   16   10   10
##  [87,] 11 66 35 20 429 445 132  51 18  66 147  68   16   26   20
##  [88,] 11 68 36 19 397 446 137  52 28  72 122  70   28   28   15
##  [89,] 12 60 36 30 421 461 101  46  5  70 183 104    3   14   56
##  [90,] 11 52 27 28 340 352  84  25  7  73 148  84    3    0   56
##  [91,]  9 62 24 12 370 323 129  24 24  85 103  46   11    4   10
##  [92,] 11 67 35 20 420 438 124  43 15  68 158  80   22   26   16
##  [93,] 11 72 37 18 451 507 152  53 17  81 129  78   25   18   16
##  [94,] 10 55 22 23 445 281 114   7  3  33 186  77   11    1   35
##  [95,] 10 45 18 27 348 199  75   8  5  28 182  46    7   28   38
##  [96,] 10 58 25 20 324 281 100  23 23  59 137  49   29   33   15
##  [97,] 10 59 26 19 324 270 115  26 37  48 125  51   31   28   15
```

```
##  [98,]  8 61 22  6 388 317 140  25 14  74  64  27   9   3   1
##  [99,] 11 64 35 20 384 400 113  48 26  66 126  69  31  33  15
## [100,] 10 57 26 19 332 302  89  26 23  73 135  57  15  16  21
## [101,] 11 67 33 22 432 401 135  42 24  56 159  68  31  35  20
## [102,] 10 57 24 21 375 335 109  22  8  66 138  63  12   0  35
## [103,] 10 59 26 19 353 298  98  26 24  67 127  52  29  26  16
## [104,] 10 42 17 28 315 209  64   8  4  40 155  56   3   0  56
## [105,] 12 71 42 24 434 569 120  64 12 105 166 108  11  12  36
## [106,] 10 39 15 30 309 174  62   6  4  29 167  44   1  14  56
## [107,] 11 68 33 22 420 430 124  42 14  75 160  68  31  35  20
## [108,] 10 61 26 19 354 313 112  26 25  70 132  53  27  24  17
## [109,] 10 49 19 26 297 181  80   7 20  33 140  50  40  31  32
## [110,] 11 48 26 29 333 288  78  16 10  43 154  93   9   0  56
## [111,] 11 56 27 28 426 355 111  25  8  48 179  84   5   0  56
## [112,] 11 58 29 26 304 322  83  31 27  77 144  66  30  36  32
## [113,] 10 65 32 13 395 372 130  44 32  58  95  56  27  12   8
## [114,] 10 53 25 20 325 233  89  22 25  31 132  51  31  32  15
## [115,] 11 54 25 30 401 333 102  21  4  53 189  74   5  14  56
```

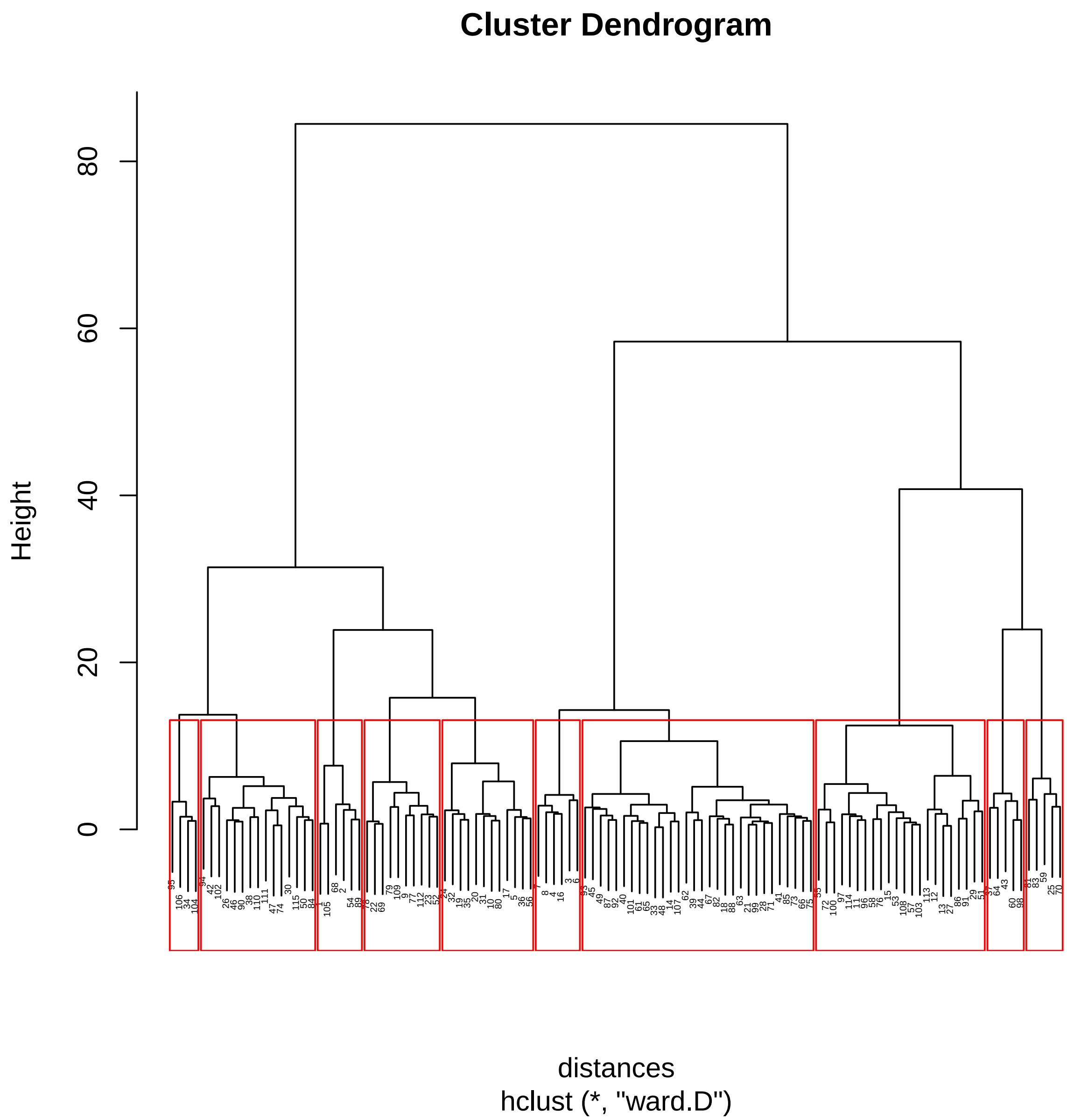
```r
#
```

3.2 H-Clustering

I use the euclidian distance for hierarchical clustering and made the cut to form ten (10) partitions.

```r
# heatmap(net_orbits_scaled, cexRow = 0.3)

# Ward Hierarchical Clustering
distances  <- dist(net_orbits_scaled, method = "euclidean") # distance matrix
hclust_fit <- hclust(distances, method="ward.D")
plot(hclust_fit, cex = 0.3) # display dendogram
k = 10
# draw dendogram with red borders around the k clusters
rect.hclust(hclust_fit, k = k, border="red")
```




```
clusters2 <- cutree(hclust_fit, k = k) # cut tree into k clusters
```

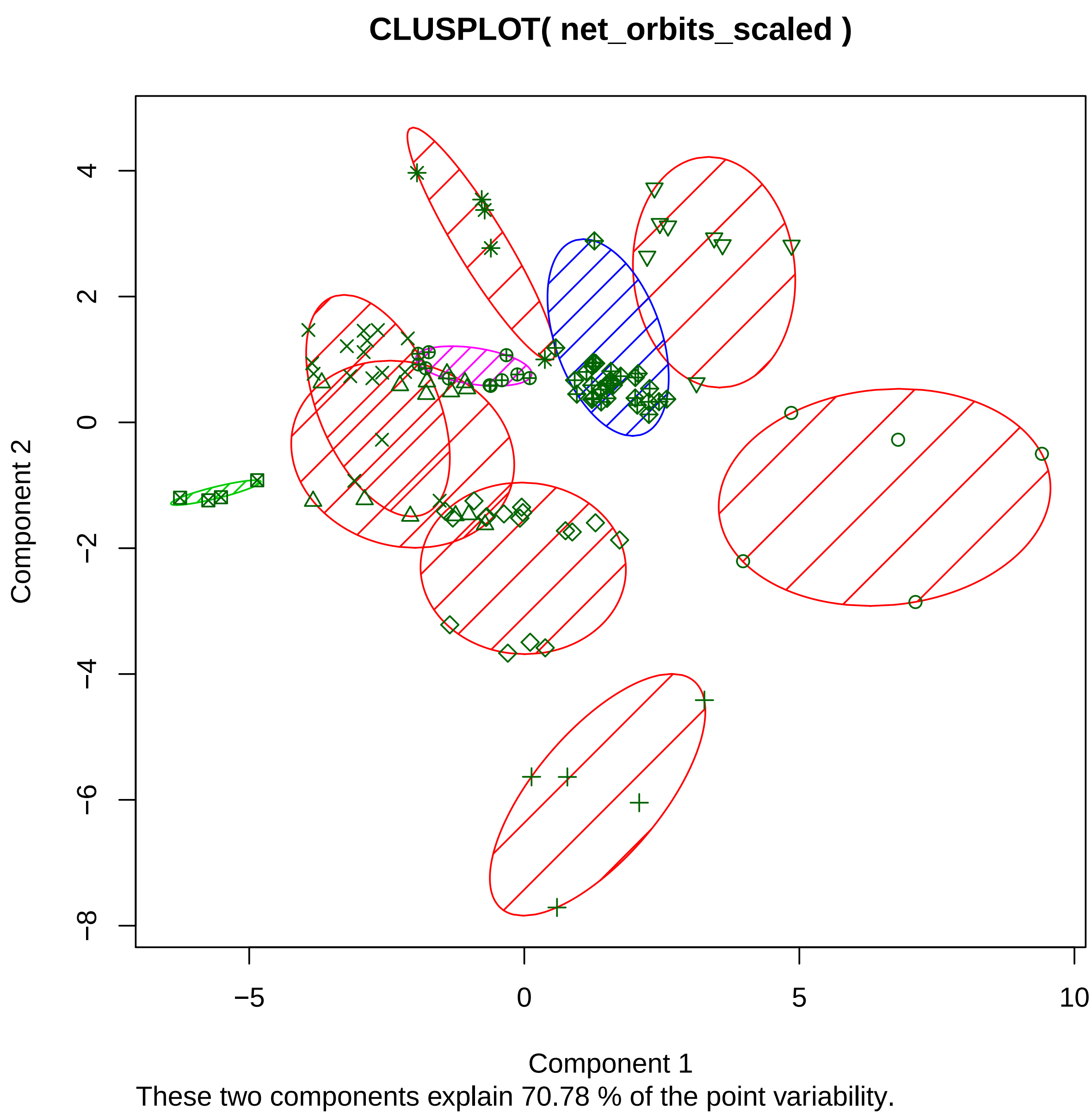
3.3 k-Means Clustering

The same number of centers (10) was used for k-Means clustering to check later if results of two different clustering methods will match.

```
k = 10

kmeans_fit <- kmeans(net_orbits_scaled, k)

# Cluster Plot against 1st 2 principal components
library(cluster)
clusplot(net_orbits_scaled, kmeans_fit$cluster, color = TRUE, shade = TRUE, labels = FALSE, lines=0)
```




```
# Centroid Plot against 1st 2 discriminant functions
#library(fpc)
#plotcluster(net_orbits_scaled, kmeans_fit$cluster)
```

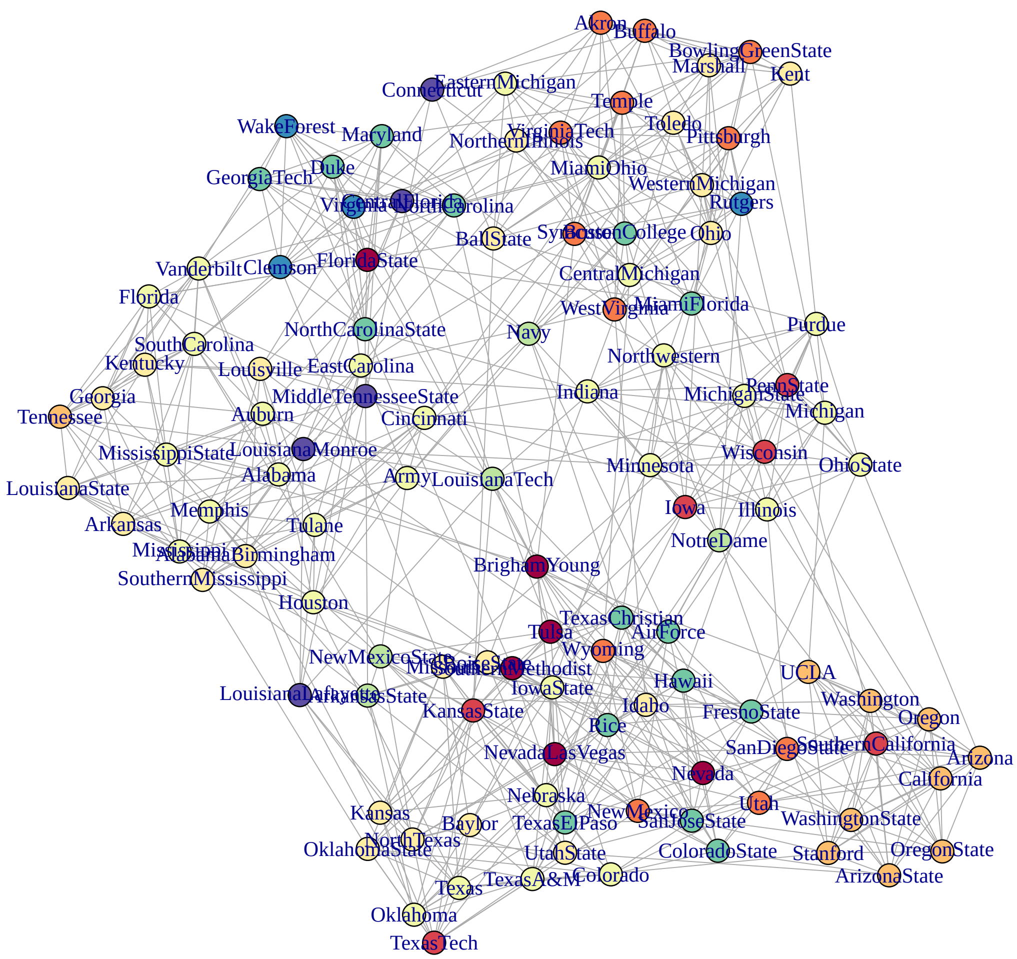
3.4 Visualization (Hierarchical Clustering)

The network coloured based on the clusters defined during hierarchical clustering.

```
# color the nodes
pal = colorRampPalette(brewer.pal(11,"Spectral"))(length(unique(clusters2)))
node_colors <- pal[clusters2]
```

```r
V(net)$color <- node_colors

#coords_fr = layout.fruchterman.reingold(net, weights=E(net)$weight)
# igraph plot options
igraph.options(vertex.size=5, edge.width=0.75)
# plot network
plot(net, layout=coords_fr, vertex.color=V(net)$color)
```

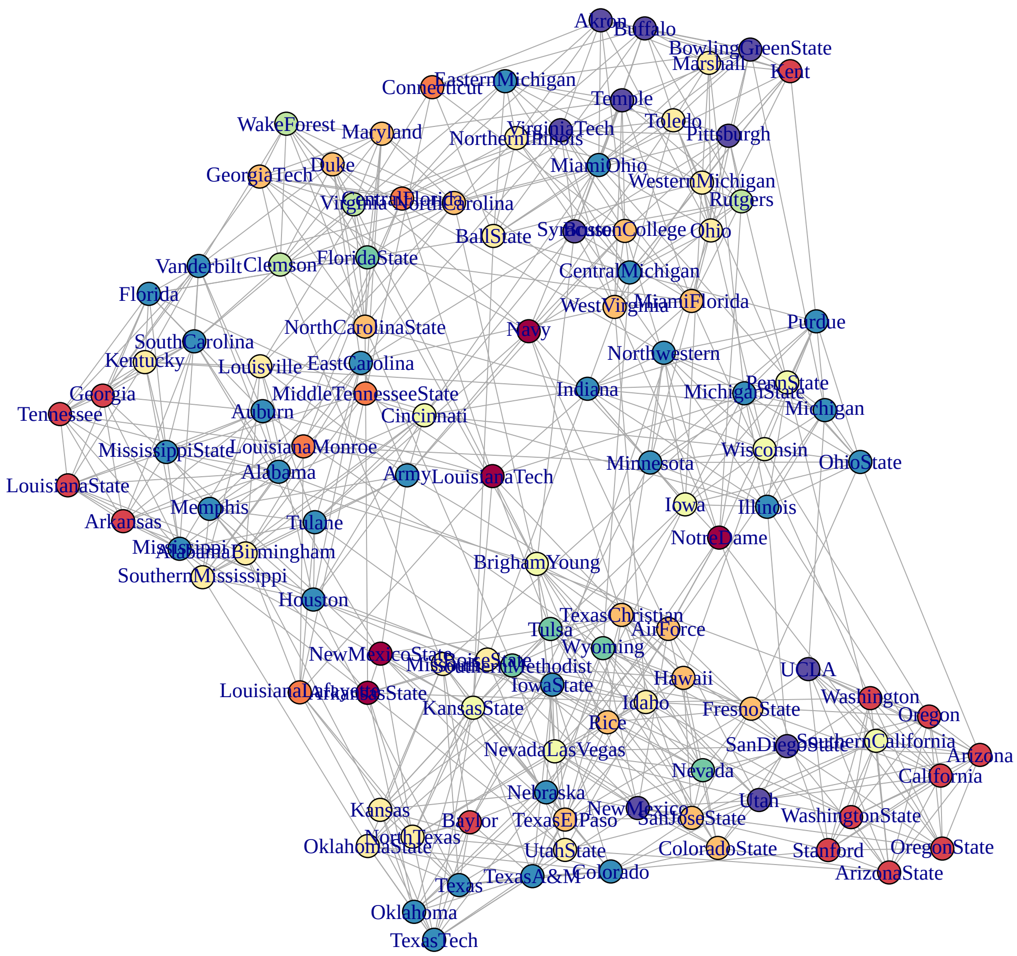


3.5 Visualization (k-Means)

The network coloured based on the clusters defined during k-Means clustering.

```r
# color the nodes
pal = colorRampPalette(brewer.pal(11,"Spectral"))(length(unique(kmeans_fit$cluster)))
node_colors <- pal[kmeans_fit$cluster]
V(net)$color <- node_colors

# coords_fr = layout.fruchterman.reingold(net, weights=E(net)$weight)
# igraph plot options
igraph.options(vertex.size=5, edge.width=0.75)
# plot network
plot(net, layout=coords_fr, vertex.color=V(net)$color)
```



4 Findings

1. Clustering based on the graphlet spectra helps finding nodes with simular set of associations in a networks (in terms of the graphlets).
2. Different clustering methods show simular results with some differences. To check that one can open the report twice in two different windows and compare the two visualizations of clustering.

References

Girvan, M., and M. E. J. Newman. 2002. "Community Structure in Social and Biological Networks." *Proceedings of the National Academy of Sciences* 99 (12). National Academy of Sciences:7821–6. https://doi.org/10.1073/pnas.122653799.