Examining Relationships Within the *Game of Thrones* Character Network

Iva Boishin, Justine Deleval and Daniela Tuiran

Introduction

The goal of this analysis is to use Exponential Random Graph Model (ERGM) and centrality measures to understand the network in the *Game of Thrones* world. We will first start with ERGM to determine the attributes that drive the edge creation and then zoom in to each of the character's position within the *Game of Thones* network.

Data

The data selected is network data for the relationship between the characters of the third novel of Game of Thrones: *A Storm of Swords*. The dataset contains the names of 107 characters of the book along a weighted relationship between those characters. This weighted relationship was calculated based on the occurrence of the two characters' names within 15 words of each other in the book. Additional data was found online to link each of those characters to their respective house allegiance.

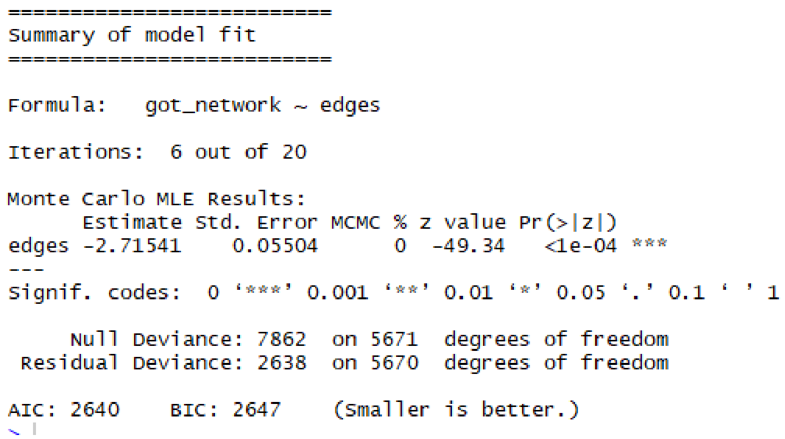
Data Preprocessing

The first step of the data preprocessing was to convert the list of the two characters names (target and source) to a matrix that will be filled out with the weight of the relationship or 0 if no relationship exists. We then created a network object using the function as.network() to be able to run the different functions. Later, we added the additional data on the house allegiance as a vertex attribute.

Model and Analysis

1- Exponential Random Graph Models

The first model that we applied is the Exponential Random Graph Models (ERGM). The goal of this model is the identify the processes that influence the creation of links (edges). We first decided to estimate a simple model that only examine the edge density by calling the function ergm() on the network object previously created. Below are the results found:

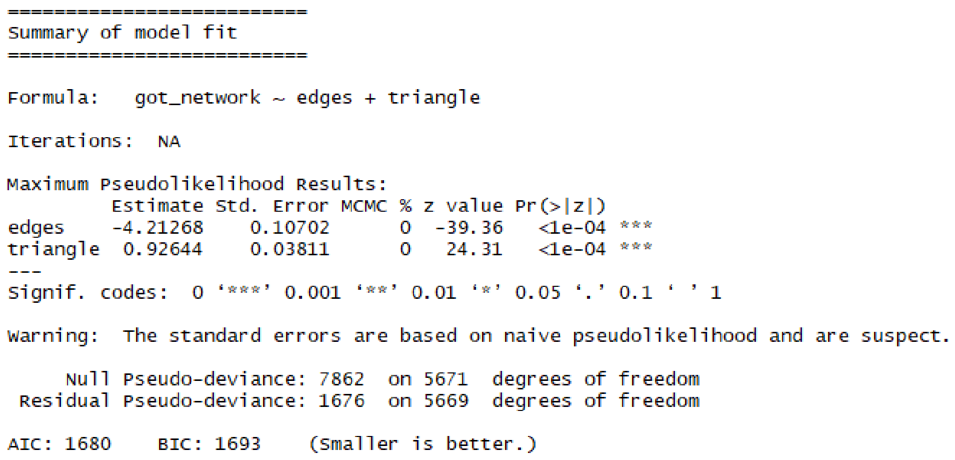


```
==========================
Summary of model fit
==========================

Formula:   got_network ~ edges

Iterations:  6 out of 20

Monte Carlo MLE Results:
      Estimate Std. Error MCMC % z value Pr(>|z|)
edges -2.71541    0.05504      0  -49.34   <1e-04 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

     Null Deviance: 7862  on 5671  degrees of freedom
 Residual Deviance: 2638  on 5670  degrees of freedom

AIC: 2640    BIC: 2647    (Smaller is better.)
```

We get an edge parameter of –2.71. It is important to note that the negative edge parameter confirms that fact that our network is rather sparse. The edge parameter is calculated as the log of the edge odds. We can thereafter calculate the corresponding probability that an edge will be created between pairs of nodes. By computing the exponential of the edge parameter divided by one plus the exponential of this same edge parameter, it gives us a probability of 6% that an edge will be created.

To improve this base model, we tried to use the *triangle* signature in our model, which takes into account the completed triangles, but the model did not converge. As a result, we changed the estimate measure from Monte Carlo maximum likelihood (MCMLE) to maximum pseudolikelihood estimation (MPLE). This is used whenever a network is too large and needs to be estimated instead of calculated precisely in order to make it computationally feasible[1]. It is important to note that the drawback of this approach is that it generally underestimates standard errors. We ran the ERGM function with this estimate specification both on the base edge model as well as on a model using a vertex attribute (mentioned below) and obtained the exact same results with MPLE as with MCMLE. Thus, it seems like for this network, this is actually a nonissue.

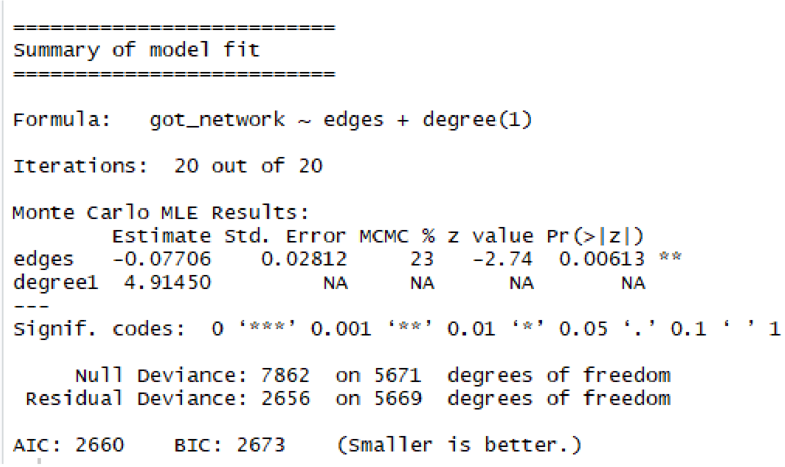
[1] Cornell University: https://arxiv.org/abs/1708.02598



```
==========================
Summary of model fit
==========================

Formula:   got_network ~ edges + triangle

Iterations:  NA

Maximum Pseudolikelihood Results:
         Estimate Std. Error MCMC % z value Pr(>|z|)    
edges    -4.21268    0.10702      0  -39.36   <1e-04 ***
triangle  0.92644    0.03811      0   24.31   <1e-04 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Warning:  The standard errors are based on naive pseudolikelihood and are suspect.

     Null Pseudo-deviance: 7862  on 5671  degrees of freedom
 Residual Pseudo-deviance: 1676  on 5669  degrees of freedom
 
AIC: 1680    BIC: 1693    (Smaller is better.)
```

From this summary, we can see that the triangle signature is a strong driving force of this network, both from the significant p-value but more importantly from the drastic accuracy improvement shown by the AIC and BIC measures.

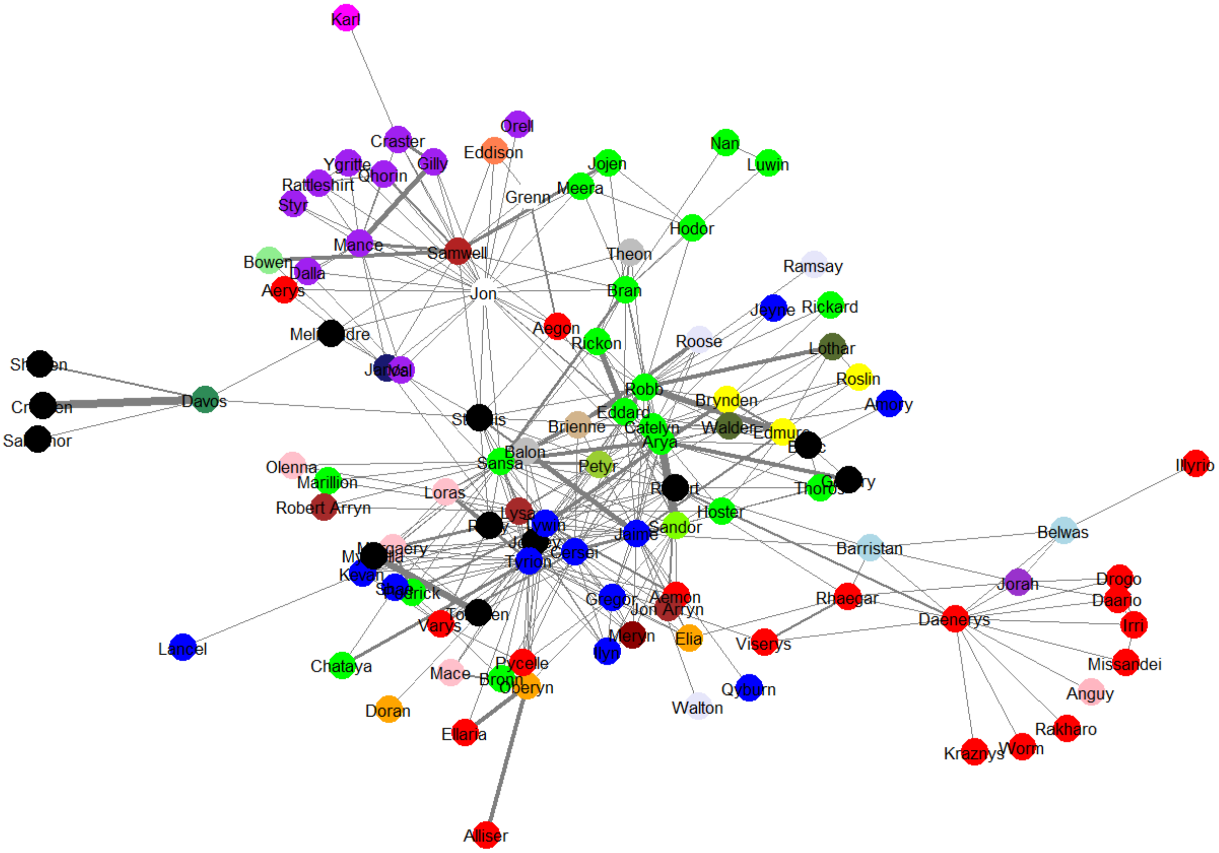
We can also run ERGM with another signature that is the *degree*() non-parametric attribute. The degree represents the frequency distribution for nodal degrees in which each node counts only once. We used degree(1), in which 1 represent a vector of distinct integers. It adds one network statistic to the model for each element. Running this model, we get AIC score = 2660 and BIC score= 2673, which underperform compared to the other models.



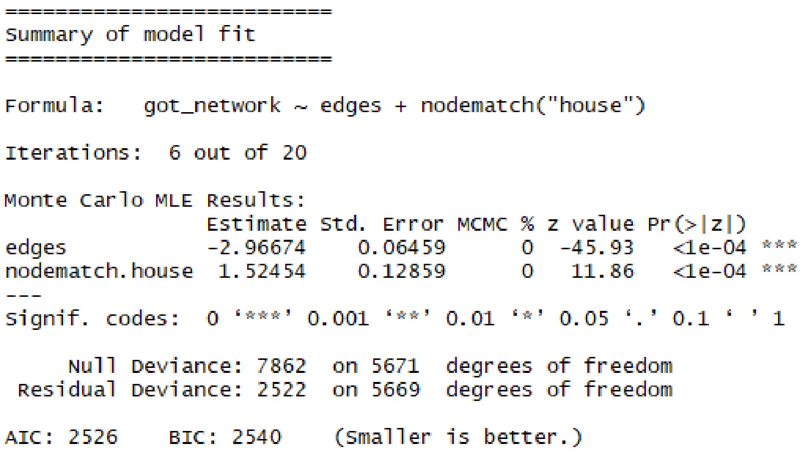
```
==========================
Summary of model fit
==========================

Formula:   got_network ~ edges + degree(1)

Iterations:  20 out of 20

Monte Carlo MLE Results:
        Estimate Std. Error MCMC % z value Pr(>|z|)   
edges   -0.07706    0.02812     23   -2.74  0.00613 **
degree1  4.91450         NA     NA      NA       NA   
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

     Null Deviance: 7862  on 5671  degrees of freedom
 Residual Deviance: 2656  on 5669  degrees of freedom
 
AIC: 2660    BIC: 2673    (Smaller is better.)
```

1-1 ERGM with House Node Attributes

While the triangle model significantly improved the base ERGM performance, we wanted to see if the model could be further improved. Looking online, we found information relating to the house allegiance that each of the characters has. Plotting that house allegiance shows clear communities within the network. The House of Targaryen (in red) and the FreeFolk (in purple) seem to be very connected within each other. Unlike the Free Folk, the House of Targaryen is rather disconnected with the rest of the characters, except for the few Targaryen characters that are rather disconnected from their house. On the other hand, the House of Stark (in light green) is more or less disconnected from each other, while also being in the center of the entire population. It is also interesting to note that the House of Tully (in yellow) and the House of Lannister (in blue) are rather connected amongst each other but also within the rest of the population as well.



From these insights, we thought that it could be interesting to see if a random model taking into account the houses would generate a more accurate ERGM. We applied a *nodematch* signature with the categorical house vertex attribute and obtained the results below.

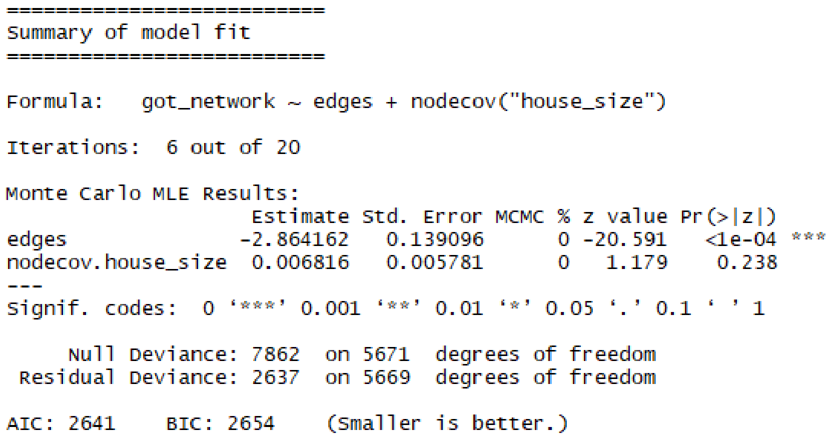

```
==========================
Summary of model fit
==========================

Formula:   got_network ~ edges + nodematch("house")

Iterations:  6 out of 20

Monte Carlo MLE Results:
                Estimate Std. Error MCMC % z value Pr(>|z|)    
edges           -2.96674    0.06459      0  -45.93   <1e-04 ***
nodematch.house  1.52454    0.12859      0   11.86   <1e-04 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

     Null Deviance: 7862  on 5671  degrees of freedom
 Residual Deviance: 2522  on 5669  degrees of freedom
 
AIC: 2526    BIC: 2540    (Smaller is better.)
```

As given by the p-value for the *nodematch.house* coefficient, the house allegiance is in fact a signature that is driving this network. The fact that the estimate of the coefficient is positive means that we see more configurations linking nodes from the same house than we would expect to see by chance. The better fit of this model is also reflected in the decreased accuracy values from the model that only considers edge density (AIC = 2640 and BIC = 2647). (Note: We verified the MPLE estimate method with this model.)

Additionally, we ran a nodemix model to see if some characters are more likely to have interactions because they come from a certain pair of houses. This turned out to not be the case. The only house pairs that had a significant p-value had a coefficient of –Inf while the house pairs that did not have a coefficient of –Inf did not have a significant p-value. Moreover, the accuracy measures of the resulting model are significantly worse than any of the other models at AIC = 6083 and BIC = 8421.

We also wanted to see if belonging to a certain house increased a character's chances of forming relationships. From the model summary, we saw that this was indeed the case for some of the characters (see Appendix 1). Characters belonging to houses Clegane, Lannister, Snow, Stark and Tarly seem to have more connections (i.e. edges) based on their significant p-values and positive coefficients. This was expected as those characters tend to be in the middle of the network plot as evident from the plot at the beginning of this section. It is however important to note that the House of Snow only has one character mentioned in the book and thus may not be representative of his house behavior as a whole (throughout the entire series for example). Likely due to the large number of insignificant coefficients, this model receives mixed accuracy results: AIC = 2527 and BIC = 2700.

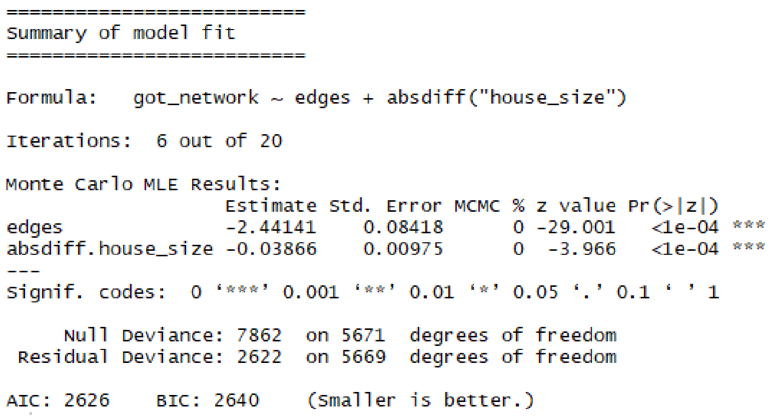
Next, we thought it might be interesting to see if the house *size* has any impact on the network model. Thus, we computed a house size attribute and tested two potential signatures. The first was to see if the characters from larger houses are more likely to form connections. The model summary returned the results below.



```
==========================
Summary of model fit
==========================

Formula:   got_network ~ edges + nodecov("house_size")

Iterations:  6 out of 20

Monte Carlo MLE Results:
                    Estimate Std. Error MCMC % z value Pr(>|z|)    
edges              -2.864162   0.139096      0 -20.591   <1e-04 ***
nodecov.house_size  0.006816   0.005781      0   1.179    0.238    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

     Null Deviance: 7862  on 5671  degrees of freedom
 Residual Deviance: 2637  on 5669  degrees of freedom
 
AIC: 2641    BIC: 2654    (Smaller is better.)
```

From the model summary, we can see that the coefficient for house size *nodecov* is not significant. As a result, it does not seem like the house size attribute is a signature that is driving this network. This is further confirmed by the fact that the AIC and BIC accuracy measures are slightly worse than those of the base model including only the edge density (AIC = 2640 and BIC = 2647).

The second model was to see whether characters from similar house sizes have a tendency to form relationships with each other. While the coefficient for this variable is significant, the fact that the coefficient is negative shows that there is an inverse relationship. Given the magnitude of the score though, the relationship does not seem to be very strong. This is reflected in the fact that the accuracy measures were only very slightly improved.

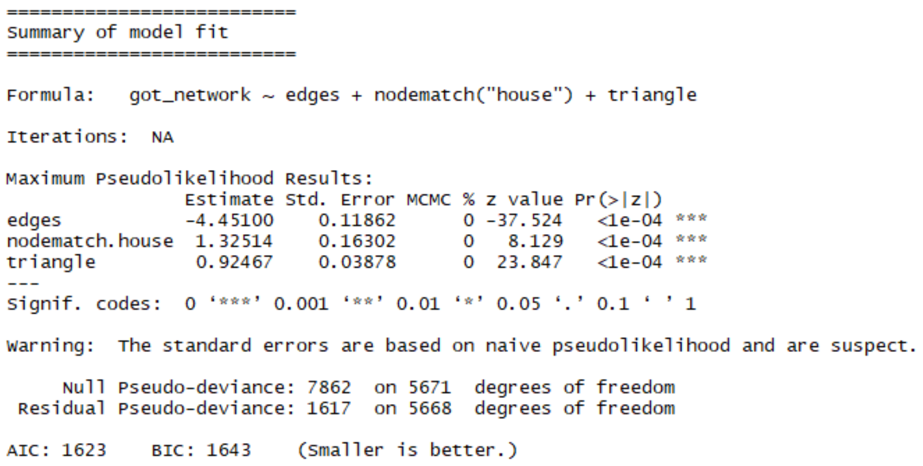


```
==========================
Summary of model fit
==========================

Formula:   got_network ~ edges + absdiff("house_size")

Iterations:  6 out of 20

Monte Carlo MLE Results:
                   Estimate Std. Error MCMC % z value Pr(>|z|)    
edges              -2.44141    0.08418      0 -29.001   <1e-04 ***
absdiff.house_size -0.03866    0.00975      0  -3.966   <1e-04 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

     Null Deviance: 7862  on 5671  degrees of freedom
 Residual Deviance: 2622  on 5669  degrees of freedom
 
AIC: 2626    BIC: 2640    (Smaller is better.)
```

In the end, it seems like the *nodematch* with the categorical house vertex attribute has the most predictive power. It would then be interesting to combine this vertex signature with the triangle signature to see if the model can be further improved.

1-2 ERGM Vertex Attribute with Triangle

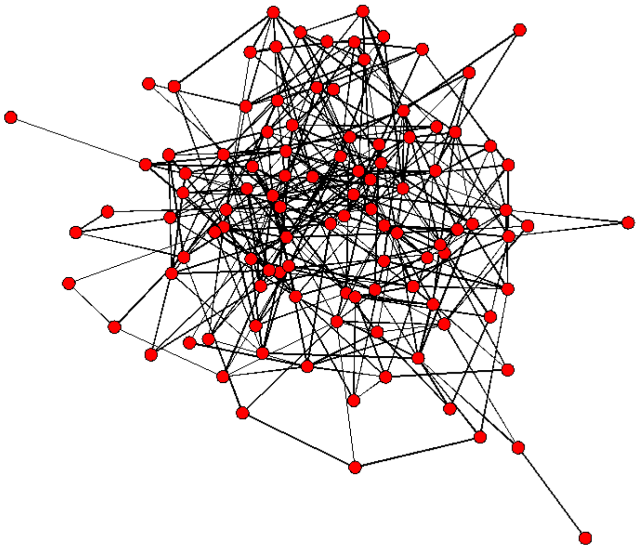
As can be seen from the significant p-values, both signatures are important in determining the edge creation for this network. This combined model outperforms the ERGM with only edge density and triangle attributes (which had accuracy measures of AIC: 1680 and BIC = 1693). After that, we simulated ten networks using this model.



```
==========================
Summary of model fit
==========================

Formula:   got_network ~ edges + nodematch("house") + triangle

Iterations:  NA

Maximum Pseudolikelihood Results:
                Estimate Std. Error MCMC % z value Pr(>|z|)    
edges           -4.45100    0.11862      0 -37.524   <1e-04 ***
nodematch.house  1.32514    0.16302      0   8.129   <1e-04 ***
triangle         0.92467    0.03878      0  23.847   <1e-04 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Warning:  The standard errors are based on naive pseudolikelihood and are suspect.

     Null Pseudo-deviance: 7862  on 5671  degrees of freedom
 Residual Pseudo-deviance: 1617  on 5668  degrees of freedom
 
AIC: 1623    BIC: 1643    (Smaller is better.)
```

1-3 Simulated ERGM

With the *nodematch* and triangle ERGM, we get a probability distribution across networks of the same size with the same density of triangle. If the simulated network resembles the observed data, it is a good indication that the model is a good fit to the observed data. Below is one simulated network that was build using the simulate() function on the ERGM fit. It is important to note that each time that the ERGM function is run, the resulting model will change as it is simply a random model with a set probability of edge formation and thus depends on the seed. Even the ten plots created during this process all looked significantly different. This is because the function is simply flipping a biased coin based on the edge density (as the probability) to test whether each pair of nodes will have a connection or not. Below is one of the network plots that resulted from running this function. These types of plots are mostly used to examine the network structure and the relationship between the various actors in a network, without having sensitive information about the individual actors. This is most often used in the health industry.



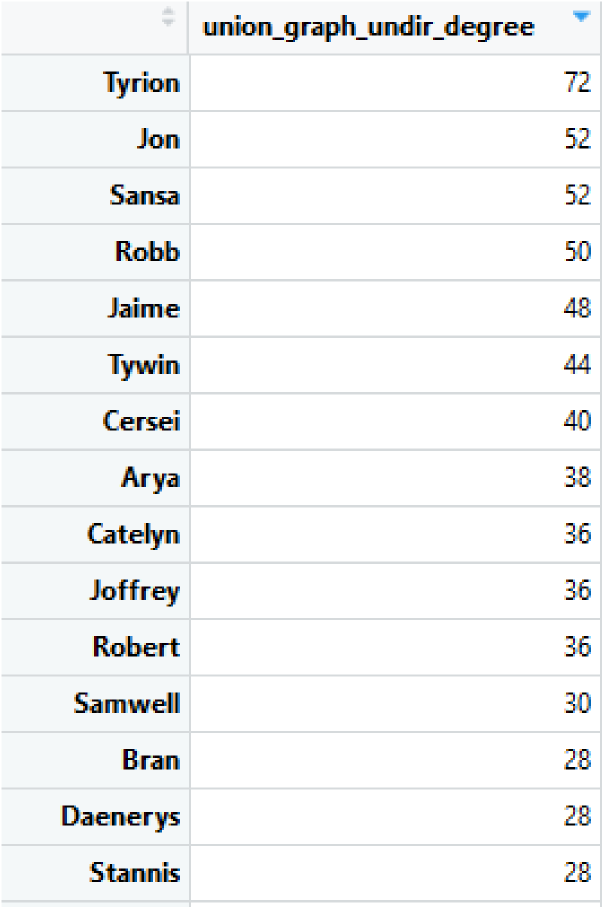
2- Centrality measures

The centrality measures are used to identify the most embedded vertices of a network. Because there is no consensus on exactly how centrality is defined, we will used different method of centrality measure for our analysis of the relationship between the different characters of the book. Notably, we will examine degree, closeness, betweeness and eigenvector centralities.

2-1 Degree

The degree centrality is simply defined has the number of links that a node has. In a directed network, it tells us how many in- and out-going edges a node has. Since this network is undirected, the degree centrality measures the number of edges a node has. Because these edges are weighted, it is actually a sum of the weights as opposed to a simple count. Tyrion is the character with the highest weighted connections in the network. This makes sense given the fact that in this book Tyrion is forced to marry Sansa Stark so that the Lannister can claim Winterfell and gain power in the north.

We can also say that each one of the characters with higher node degree represents an important family in the seven Kingdoms, so their role is crucial for the development of the story.

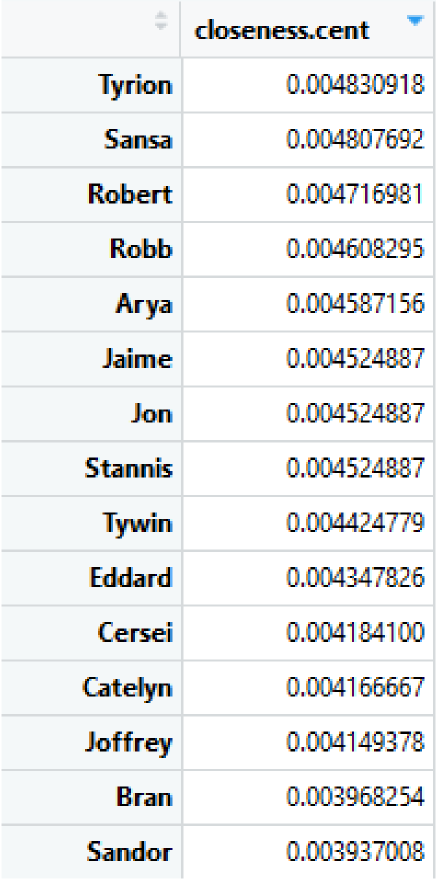


	union_graph_undir_degree
Tyrion	72
Jon	52
Sansa	52
Robb	50
Jaime	48
Tywin	44
Cersei	40
Arya	38
Catelyn	36
Joffrey	36
Robert	36
Samwell	30
Bran	28
Daenerys	28
Stannis	28

2-2 Closeness

The closeness of a node is measure as the average length of the shortest path between this node and all the other nodes of the graph. Therefore, the nodes that are more central, are also closer to all the other nodes. This centrality measures a nodes ability to affect the other nodes in the network.

The characters with highest closeness all surround central characters that connect various storylines and houses in Game of Thrones. Comparing the results with the node degree, we can see that using the closeness measure change the centrality of some of the characters. It appears that Tyrion is still the character high the highest centrality (highest closeness score), but Jon only appears at the 7th place. This suggests that while Jon may have the second largest number of direct weighted connections, he has relatively weaker reach within the network as a whole. Additionally, Eddard Stark appears in the ranking of the top 15 characters using the closeness measures which could be due to the fact that he has a big family with a lot of children with whom he interacts often.

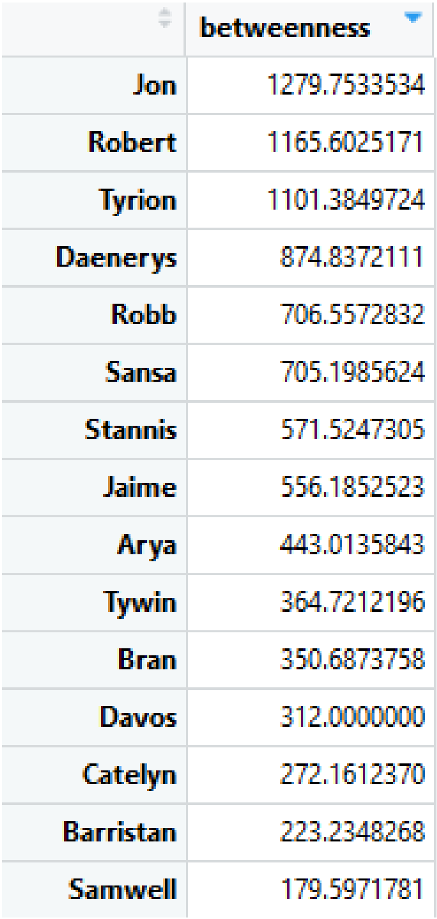


	closeness.cent
Tyrion	0.004830918
Sansa	0.004807692
Robert	0.004716981
Robb	0.004608295
Arya	0.004587156
Jaime	0.004524887
Jon	0.004524887
Stannis	0.004524887
Tywin	0.004424779
Eddard	0.004347826
Cersei	0.004184100
Catelyn	0.004166667
Joffrey	0.004149378
Bran	0.003968254
Sandor	0.003937008

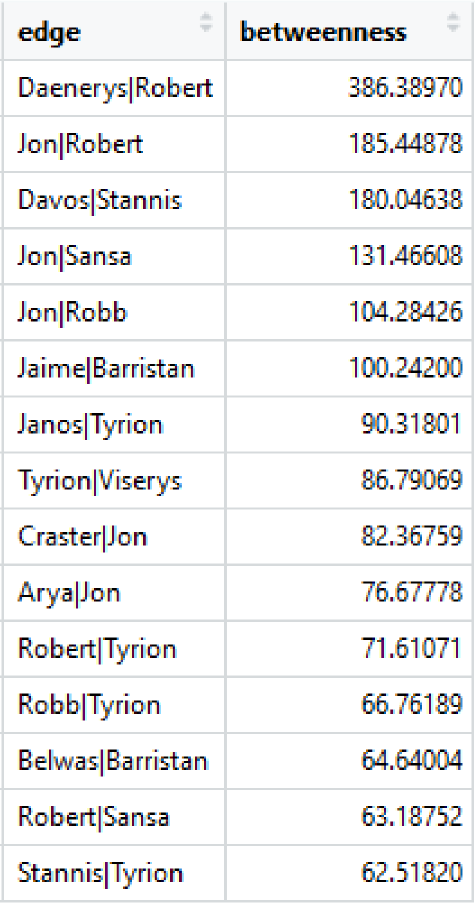
2-3 Betweenness

The measure of betweenness tell us the number of times a node is in the shortest path between two other nodes. If a node has a high betweenness centrality, it means that it is on the path between many other nodes and is likely an important connector in the network. Usually, those characters with high betweenness will be key element of the network as they are linked to many other characters and able to communicate to wider range of characters. Using the betweenness measure, it seems like Jon becomes the most central character. Looking at the network graph above, we can see that Jon is in fact a very important link between the Free Folk and the rest of the characters. In this ranking, Tyrion appears third because he is still very central as he is a bridge between the Lannister and the Starks.

Regarding edge betweenness, the connections between Daenerys and Robert is the most important in the network as that is the link that connects the Targaryen characters to the rest of the network. It is not surprising that these characters are often mentioned together as they represent the two sides interested in taking the Iron throne. On one side there is Robert with the Lannister's family and all its allies, and on the other side its Daenerys claiming her birth right to the throne with the opposition to the actual King.



	betweenness
Jon	1279.7533534
Robert	1165.6025171
Tyrion	1101.3849724
Daenerys	874.8372111
Robb	706.5572832
Sansa	705.1985624
Stannis	571.5247305
Jaime	556.1852523
Arya	443.0135843
Tywin	364.7212196
Bran	350.6873758
Davos	312.0000000
Catelyn	272.1612370
Barristan	223.2348268
Samwell	179.5971781

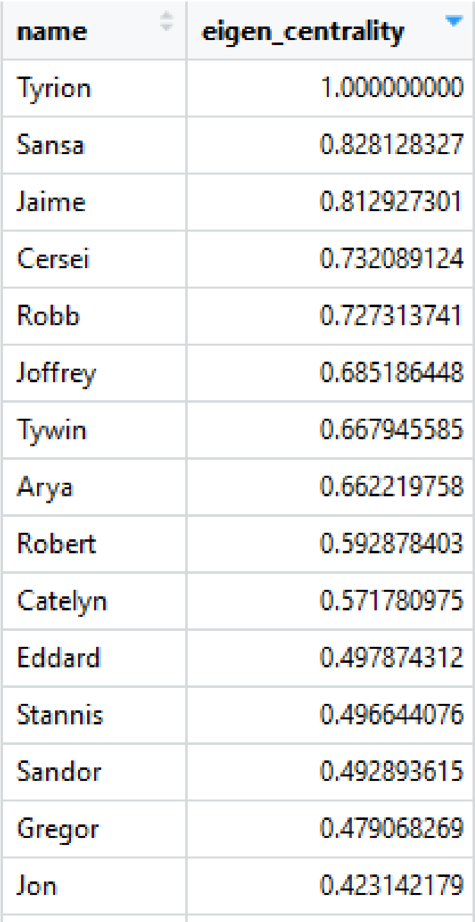


edge	betweenness
Daenerys|Robert	386.38970
Jon|Robert	185.44878
Davos|Stannis	180.04638
Jon|Sansa	131.46608
Jon|Robb	104.28426
Jaime|Barristan	100.24200
Janos|Tyrion	90.31801
Tyrion|Viserys	86.79069
Craster|Jon	82.36759
Arya|Jon	76.67778
Robert|Tyrion	71.61071
Robb|Tyrion	66.76189
Belwas|Barristan	64.64004
Robert|Sansa	63.18752
Stannis|Tyrion	62.51820

2-4 Eigenvector

The eigenvector approach to centrality calculates a relative score for each node in the network based on the idea that connections to high-scoring nodes contribute more to the score of the node in question than equal connections to low scoring nodes. This shows the importance of a node given the importance of its friends. For this reason, this centrality measure is often referred to as the prestige centrality.

Taken together, we could say that House Stark (specifically Robb and Sansa) and House Lannister have the most important connections in Game of Thrones. Tyrion is again the character with most important connections within the network. This means that Tyrion is not only the character with the highest number of weighted connections (betweenness) but also the one with the most connections to prestigious characters. Tyrion, Jaime, Cersei and Joffrey belong to the same family, this is an indicator of the power or importance of the Lannister in Westeros. It shows that they are connected to important characters in the Game of Thrones network.



name	eigen_centrality
Tyrion	1.000000000
Sansa	0.828128327
Jaime	0.812927301
Cersei	0.732089124
Robb	0.727313741
Joffrey	0.685186448
Tywin	0.667945585
Arya	0.662219758
Robert	0.592878403
Catelyn	0.571780975
Eddard	0.497874312
Stannis	0.496644076
Sandor	0.492893615
Gregor	0.479068269
Jon	0.423142179

Discussion, conclusion and interpretation of main insights

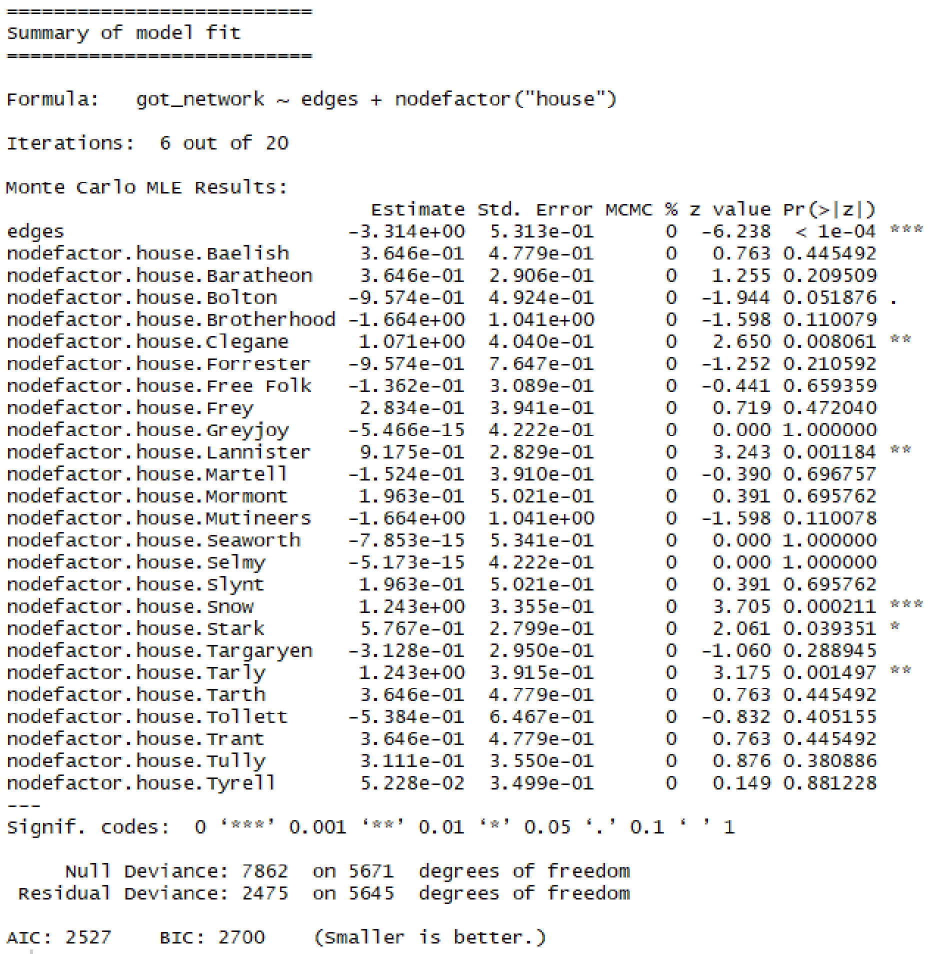
The purpose of ERGM, is to describe the local selection forces that shape the global structure of a network. By fitting different ERGM models and varying the arguments and interactions present in each one, we can understand the effects of different social network attributes in the explanation of the relationship of the different nodes and the edge creation in the network. This can also be useful for the estimation of changes or impacts that might occur and the causes or sources of them. In this way, ERGM can help to understand a network singularity or to simulate new random realizations of networks that retain the essential properties of the original.

Centrality measures are extremely important in quantifying the embeddedness of the characters of Games of Thrones in the network. Looking at the overall results of the centrality measures, it appears that the main characters, such as Tyrion or Jon Snow, have the highest centrality scores when using different metrics of analysis. We can also notice that the top 15 characters of each centrality measures almost always included characters that play a big role in the story of Game of Thrones. Nevertheless, there were differences in the rankings as they all measure different constructs of centrality.

The different applications for social network analysis in the Game of Thrones data, gives us some interesting and accurate results that are also aligned with what is really happening in the story. It demonstrates the power of social network analysis and the further applications that can be used in businesses to understand the complexity of real-life social networks. This analysis will help organizations in understanding networks (ex. their customer base), which will help in making better decisions. One such decision could be using centrality in determining who to send information and samples to in order to generate the most buzz for the product.

Appendix

Appendix 1 – ERGM NodeMix House Vertex Attribute model



```
==========================
Summary of model fit
==========================

Formula:   got_network ~ edges + nodefactor("house")

Iterations:  6 out of 20

Monte Carlo MLE Results:
                               Estimate Std. Error MCMC % z value Pr(>|z|)    
edges                        -3.314e+00  5.313e-01      0  -6.238  < 1e-04 ***
nodefactor.house.Baelish      3.646e-01  4.779e-01      0   0.763 0.445492    
nodefactor.house.Baratheon    3.646e-01  2.906e-01      0   1.255 0.209509    
nodefactor.house.Bolton      -9.574e-01  4.924e-01      0  -1.944 0.051876 .  
nodefactor.house.Brotherhood -1.664e+00  1.041e+00      0  -1.598 0.110079    
nodefactor.house.Clegane      1.071e+00  4.040e-01      0   2.650 0.008061 ** 
nodefactor.house.Forrester   -9.574e-01  7.647e-01      0  -1.252 0.210592    
nodefactor.house.Free Folk   -1.362e-01  3.089e-01      0  -0.441 0.659359    
nodefactor.house.Frey         2.834e-01  3.941e-01      0   0.719 0.472040    
nodefactor.house.Greyjoy     -5.466e-15  4.222e-01      0   0.000 1.000000    
nodefactor.house.Lannister    9.175e-01  2.829e-01      0   3.243 0.001184 ** 
nodefactor.house.Martell     -1.524e-01  3.910e-01      0  -0.390 0.696757    
nodefactor.house.Mormont      1.963e-01  5.021e-01      0   0.391 0.695762    
nodefactor.house.Mutineers   -1.664e+00  1.041e+00      0  -1.598 0.110078    
nodefactor.house.Seaworth    -7.853e-15  5.341e-01      0   0.000 1.000000    
nodefactor.house.Selmy       -5.173e-15  4.222e-01      0   0.000 1.000000    
nodefactor.house.Slynt        1.963e-01  5.021e-01      0   0.391 0.695762    
nodefactor.house.Snow         1.243e+00  3.355e-01      0   3.705 0.000211 ***
nodefactor.house.Stark        5.767e-01  2.799e-01      0   2.061 0.039351 *  
nodefactor.house.Targaryen   -3.128e-01  2.950e-01      0  -1.060 0.288945    
nodefactor.house.Tarly        1.243e+00  3.915e-01      0   3.175 0.001497 ** 
nodefactor.house.Tarth        3.646e-01  4.779e-01      0   0.763 0.445492    
nodefactor.house.Tollett     -5.384e-01  6.467e-01      0  -0.832 0.405155    
nodefactor.house.Trant        3.646e-01  4.779e-01      0   0.763 0.445492    
nodefactor.house.Tully        3.111e-01  3.550e-01      0   0.876 0.380886    
nodefactor.house.Tyrell       5.228e-02  3.499e-01      0   0.149 0.881228    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

     Null Deviance: 7862  on 5671  degrees of freedom
 Residual Deviance: 2475  on 5645  degrees of freedom
 
AIC: 2527    BIC: 2700    (Smaller is better.)
```