JACOBS UNIVERSITY

Advanced Project 1

Functional Interpretation of Disease-Associated Genes

Merint Thomas Mathew

In this project, we create a data analysis pipeline based on a systematic network-based approach to implement a mathematical model defined by Menche, J, et al[1]. This mathematical model helps in determining the relationship between any two disease pairs and helps us identify if there are any functional and biological similarities between the two diseases. In this report, we will demonstrate how this is achieved with the protein-based positions of two diseases in a network.

Spring 2019

Under the guidance of
Prof. Dr. Marc-Thorsten Hütt

JACOBS UNIVERSITY

Table of Contents

1. Introduction **3**

2. Data Description and Tools **4**

2.1 Disease-Gene Association Data 4

2.2 Protein Interaction Links Data 4

2.3 Protein-Gene mapping Data 4

2.4 Comorbidity Dataset 4

2.5 Tools used 5

3. Data Preprocessing **5**

4. Network Creation **5**

5. Model Creation and Shortest distance calculation **6**

5.1 Disease A distance 6

5.2 Disease B distance 7

5.3 Disease A-B protein pairs distance 8

5.4 Shortest distance sAB 8

6. Observations **9**

6.1 Comparing sAB values with the results in the paper 9

6.2 sAB results for Disease and Trait pair associations 10

6.3 Comparison with the Comorbidity dataset 11

7. Summary **13**

8. References **14**

9. Appendix **15**

1. Introduction

In this age of the increased population and improved healthcare needs, the study of Biological Systems is even paramount to aid such needs. In this project, we aim at building a pipeline by reproducing the mathematical model by *Menche. J*[1] based on network-based distances of diseases to determine the pathobiological relation between any two disease pairs. We try to highlight the relationship between disease pairs such that we can understand potential overlaps based protein node distances for any disease pairs. To achieve this we follow a network-based separation model through which we can calculate the shortest distances between the disease pairs based on their protein locations in the network.

Furthermore, in this paper, we will also try and compare the shortest distance values with the values obtained by *Menche. J, et al* in their work with the same disease pairs used in their observation. Also, we further compare the relation of our model with the comorbidity data used by *Park, J et al*[2]. We will look at this in detail as we progress through this report. Finally, we will also implement and study the model behavior on Phenotype (traits) pairs and observe how it behaves differently to the disease pairs.

We will be looking at all the steps involved to achieve this project in detail, however, this is just a highlight of steps that we will be following in the project

1. Obtaining various datasets such as Disease-Associated Gene dataset, Protein Interaction links dataset, Protein-gene mapping, and more.
2. Data Preprocessing, where we clean and process it to make it ready for our pipeline creation.
3. Network Creation, this is where we create our network by having appropriate nodes and links as per our consideration.
4. Network distance calculation, here we calculate the shortest distances between diseases A and B and compare the distances between A-B protein pairs.
5. Next, we will compare the values with the results from the paper and the Relative Risk values from the comorbidity dataset
6. Observation of disease pair values and phenotype pair values and inference.

2. Data Description and Tools

In this section, we will briefly go through the data used in this project and how we obtained it.

2.1 Disease-Gene Association Data

We obtain the Disease-Gene Association data from the DisGeNet Database and this dataset forms our primary basis of analysis and pipeline creation. The data consists of disease names, disease IDs, the genes associated with them, and many other details. It covers all disease areas (Mendelian, complex and environmental diseases), with special care on the integration and standardization of data, and to provide open access to knowledge of genes associated with human diseases[4]. It consists of gene associations for three categories of disease types, **diseases**, **phenotypes**, and **groups.** Each of the three types containing respective gene associations. The dataset contains roughly around 70000 observations.

2.2 Protein Interaction Links Data

A protein-protein interaction (PPI) involves two or more proteins binding together, often to carry out their biological function[6]. The Protein interaction links data is obtained from the STRING database and is primarily used for the network creation. It forms a systematic mapping of protein-protein interactions, or ‘interactome’ mapping, which has also been extended to higher organisms[3]. Such maps have revealed global topological and dynamic features of interactome networks[3]. It is with this dataset that we generate the network for consideration where the proteins form the nodes of the network. We will see this in detail later.

2.3 Protein-Gene mapping Data

This is another dataset obtained from STRING database, that primarily maps the diseased genes to associated proteins. This forms a very important mapping tool for our main disease-gene association dataset.

2.4 Comorbidity Dataset

The Comorbidity dataset is obtained from the paper by *Park, J et al*[2]. It contains a collection of disease pairs along with different statistical scores including the Relative Risk. We implement our model with the disease pairs mentioned here to observe our results. We will see this in detail as we progress in the paper.

2.5 Tools used

In this project, we have strictly used the R programming language with RStudio as our development environment and the strong development libraries that it offers. In particular, we have used the *igraph* and *cartography* libraries in this project, which helped us build the desired network and to calculate the distance between node hops. We have also used Excel sheets for data storage and for ease in loading data on the RStudio platform.

3. Data Preprocessing

Data received most of the time are raw and not orderly presented. This makes analyzing such data an arduous task. Often, we receive data that has missing values or values that are out of range. Thus, analysis of such data may not give us desired results, or to be more rightly said - accurate results[5]. Therefore, Data Preprocessing becomes a very essential step in our pipeline creation. In this section, I will highlight the steps undertaken to ready our data for the model creation.

As the first step, we mapped our disease gene association dataset with our gene-protein mapping dataset. This gives us a merged dataset with appropriate protein mapping for each diseased gene. This forms an essential step in our analysis as the proteins form the nodes of our network. We will see this in detail later. This also formed the basis from where we could filter out the three disease types according to our requirement.

The next important step in our data preprocessing, was to work with the Protein interaction links dataset. In our case, we chose to go with the protein links with a score greater than 850. This was primarily done to keep only the highly significant entries in the network and to enhance relevancy. The proteins form the nodes of the network and the edges are the connecting pairs of interacting proteins.

4. Network Creation

As mentioned previously, the proteins form the nodes in the network and connecting pairs of interacting proteins form the edges in the network. Therefore, in our case, we will be reproducing the protein interaction links dataset into a network in *R*, using the *igraph* library.

The *igraph* library provides standard function, which helps us graph the network as per our specifications[8]. Once the network as been reproduced in R, we are now ready to carry on with our model creation and evaluation. As the dataset is too extensive, it is inconvenient to present the actual network diagram, however, I have demonstrated the sample network with the below hand-drawn diagram with the numbered nodes being the proteins.

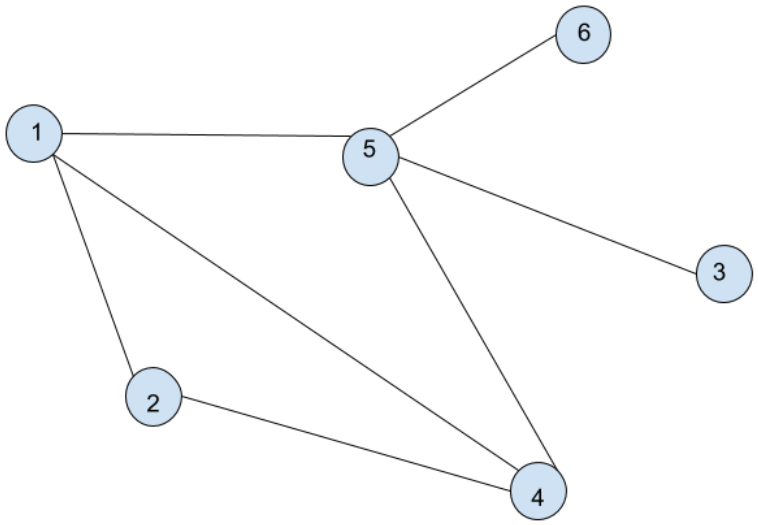

Figure 1: Random Network Diagram with labeled nodes

5. Model Creation and Shortest distance calculation

Now that we have our network ready, it's time to move towards the mathematical model implementation. We will be following the model proposed by *Menche. J[*1] in his paper which helps us understand the relationship between disease pairs. According to the model, a network-based separation between diseases A and B would help us determine whether they share similar characteristics[1]. Fundamentally, the lesser the separation score, the greater is the relationship between the two diseases. And greater is the chances of functional overlap between the two diseases. The mathematical model is given by the below formula

$$sAB = d_{\overline{AB}} - \frac{d_{\overline{AA}}+d_{\overline{BB}}}{2}$$

It compares the shortest distance between nodes of the respective diseases and the shortest distance of the nodes of one disease to the other[1]. We will observe the *sAB* values for diseases and traits in the next section. Now let's look at how we calculated the respective $d_{\overline{AA}}$, $d_{\overline{BB}}$, and $d_{\overline{AB}}$ values.

5.1 Disease A distance

The shortest distance of Disease A ($d_{\overline{AA}}$) would be the mean of the shortest path between any 2 nodes in the network for disease A. Therefore the first task here is identifying the protein nodes associated with disease A and then calculating the number of hops to obtain the shortest path between any two nodes of disease A. Let's look at Figure 2. Here we see that nodes 1, 2, and 3 are associated with disease A. And therefore, the number of hops from node 1 to node 2 is one and the number of hops from node 1 to node 3 is two. And subsequently, the number of

hops from node 2 to node 3 is three. $d_{\overline{AA}}$ calculates the mean of the 3 values and in this case, it will be 2.

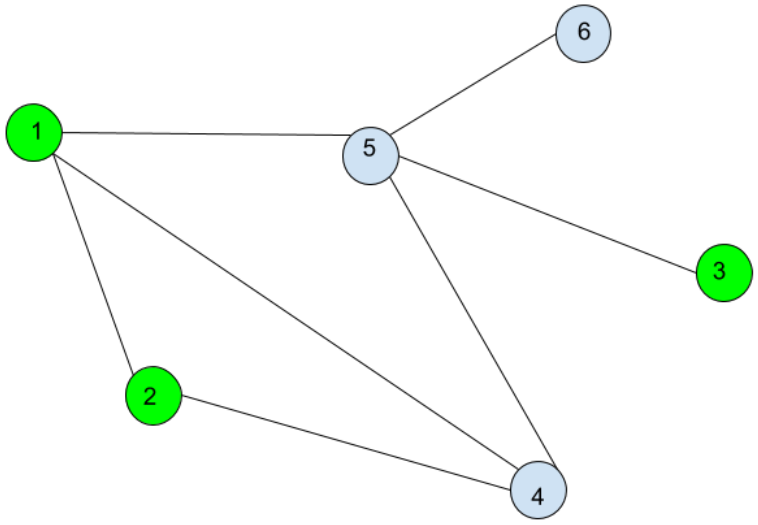

Figure 2: Network with protein nodes of Disease A highlighted in green

In our case, we use the *igraph* library once again to calculate the shortest paths between any two protein nodes for disease A. In particular, we use the *distances()* function in the *igraph*[7] library which returns a proximity matrix for all the protein nodes in disease A with its diagonal elements having the value 0. Thus, from this proximity matrix, we collect the values in the upper triangular matrix and calculate its mean. For example, in the above network, the proximity matrix turns out to be

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \end{array}\begin{bmatrix} 0 & 1 & 2 \\ 1 & 0 & 3 \\ 2 & 3 & 0 \end{bmatrix}$$

$$\begin{array}{ccc} 1 & 2 & 3 \end{array}$$

Therefore, the mean of the upper triangular matrix gives us the value 2 which is the value of $d_{\overline{AA}}$. Another thing to be wary about is the NA values that need to be eliminated.

5.2 Disease B distance

The shortest distance of Disease B ($d_{\overline{BB}}$) would be the mean of the shortest path between any 2 nodes in the network for disease B. Just like for disease A, we identify the protein nodes associated with disease B and then calculate the number of hops between 2 nodes. Let's take a look at Figure 3. Here, nodes 4, 5, and 6 are associated with disease B, where the number of hops between nodes 4 and 5 and nodes 5 and 6 is one. And the number of hops between nodes 4 and 6 is two. So the mean shortest path is 1.33.

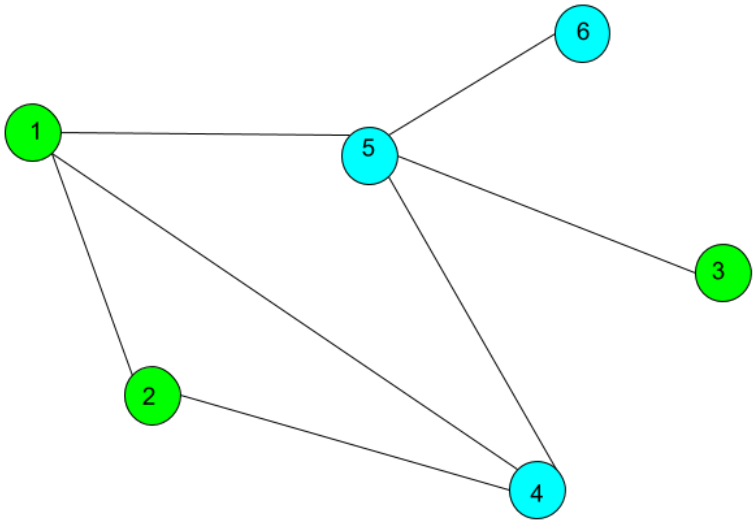

Figure 3: Network with protein nodes of Disease B highlighted in blue

Just like in disease A, in our case, we use the *distance()* function of the *igraph* library to calculate the number of hops between protein node pairs which again gives us a proximity matrix with diagonal values 0. We then take the mean of the upper triangular values to find the value of $d_{\overline{BB}}$ as demonstrated above.

5.3 Disease A-B protein pairs distance

Now that we have the individual distances of diseases A and B, we will now calculate the distance between protein nodes in disease A with the protein nodes in disease B in the same network. This can be observed in figure 3 itself and can be calculated the same way as in diseases A and B. And the mean value of these distances will give us the value of $d_{\overline{AB}}$.

In our case, we use the same *distance()* function from the *igraph* library. Just that this time, the 'to' and 'from' parameters will be nodes of disease A and nodes of disease B respectively. This will again give us a matrix of the number of hops for different node pairs. The mean of this matrix will give us the value of $d_{\overline{AB}}$.

5.4 Shortest distance *sAB*

Now that we have the values for $d_{\overline{AA}}$, $d_{\overline{BB}}$, and $d_{\overline{AB}}$, we can now calculate the *sAB* value as mentioned in the mathematical model above. This *sAB* value will help us determine the relationship between any 2 diseases A ad B. The smaller the value of *sAB*, the stronger is the relationship between the disease pairs. In fact, we could say that disease pairs with *sAB* values that are lesser than 0 have overlapping disease boundaries and share functional characteristics[1].

6. Observations

In this section, we will observe the different results that we have derived. Firstly, we have tried to find out the difference in *sAB* results in disease pairs and trait pairs. Then we have also compared the results with the sAB values obtained in the paper by *Menche. J*[1] with the same disease pairs used in the paper. And finally, we also compare our results with the same disease pairs used in the comorbidity data used by *Park, J*[2] against the relative risk value in the comorbidity dataset

6.1 Comparing *sAB* values with the results in the paper

We were able to compare the *sAB* results with the results obtained by *Menche. J*[1] for around 18000 disease pairs and we plotted a scatter plot of these values to take a look at the range that these values lie.

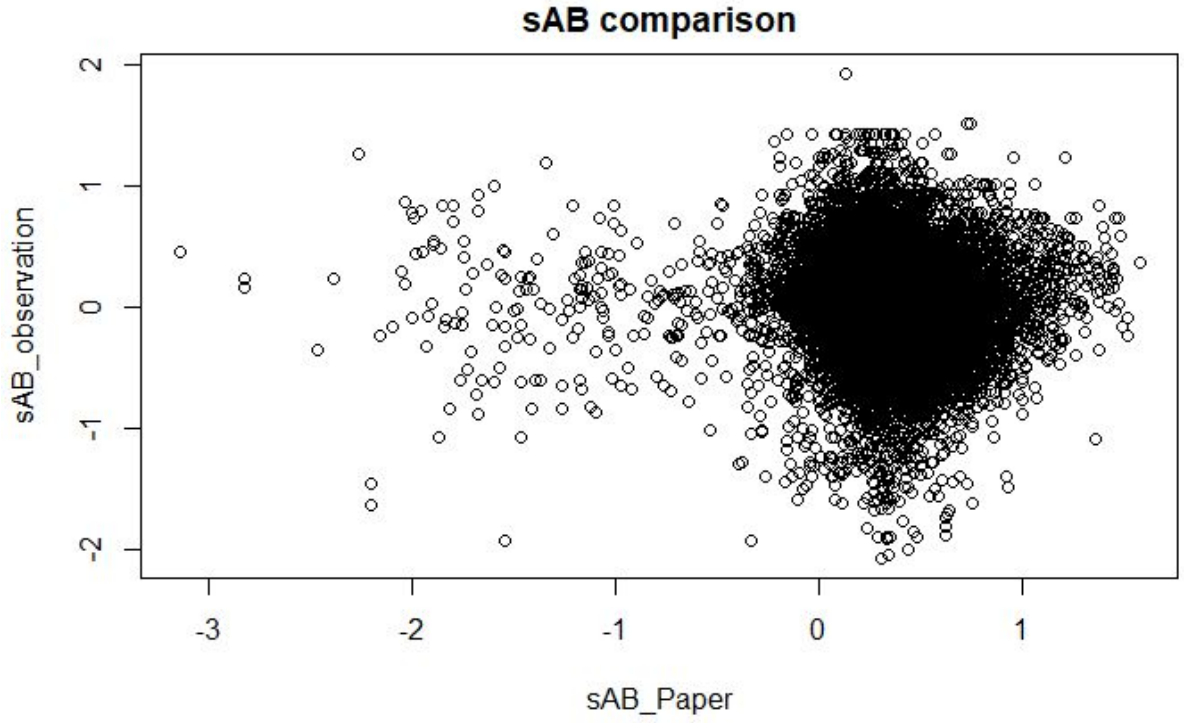

Figure 4: sAB comparison between observed values and research paper values

As we can observe, most of the values of *sAB_observed* and *sAB_paper* lie between the range of -1 to 1 for these 18000 disease pairs. What we also observe here is that there isn't a proper correlation between the 2 values. One of the major reasons for this is the dissimilarity in the disease names and disease IDs in both the datasets. In our case, we have tried to match the names of the disease pairs used in the paper with the disease mapping data and appropriate pattern matching tools in R. But this does cause some alterations in the *sAB* values from the

values obtained in the paper. However, the values do not differ much from the values from the paper which can be seen by the crowding of data points between -1 and 1.

In this case, we can investigate this hypothesis by isolating the disease pairs with extremely low *sAB* values (ideally, the values below 0 should be considered). After that, we can check for the diseased genes that are shared by both the diseases. The more the number of shared genes, the stronger is the relationship between the 2 diseases. This also means that there will be a substantial overlap in the pathobiological characteristics between the two diseases and thus, verifying our hypothesis. However, this is currently out of the scope of the project and can be left for future investigation.

6.2 *sAB* results for Disease and Trait pair associations

We have then calculated the *sAB* values for disease pairs in our disease-gene association dataset. We have evaluated the *sAB* values for around 20000 disease pairs. In this data selection, we have only chosen diseases, i.e. we have left out the other two disease types (phenotypes, groups) to observe the trend it follows.

We then followed the same procedure with trait pairs as well and evaluated the *sAB* values for around 40000 trait pairs. In this data selection, we have only chosen phenotypes, i.e. we have left out the other two disease types (diseases, groups) to observe the trend it follows.

We then compared both the results with a histogram plot for both diseases and traits and observed the following

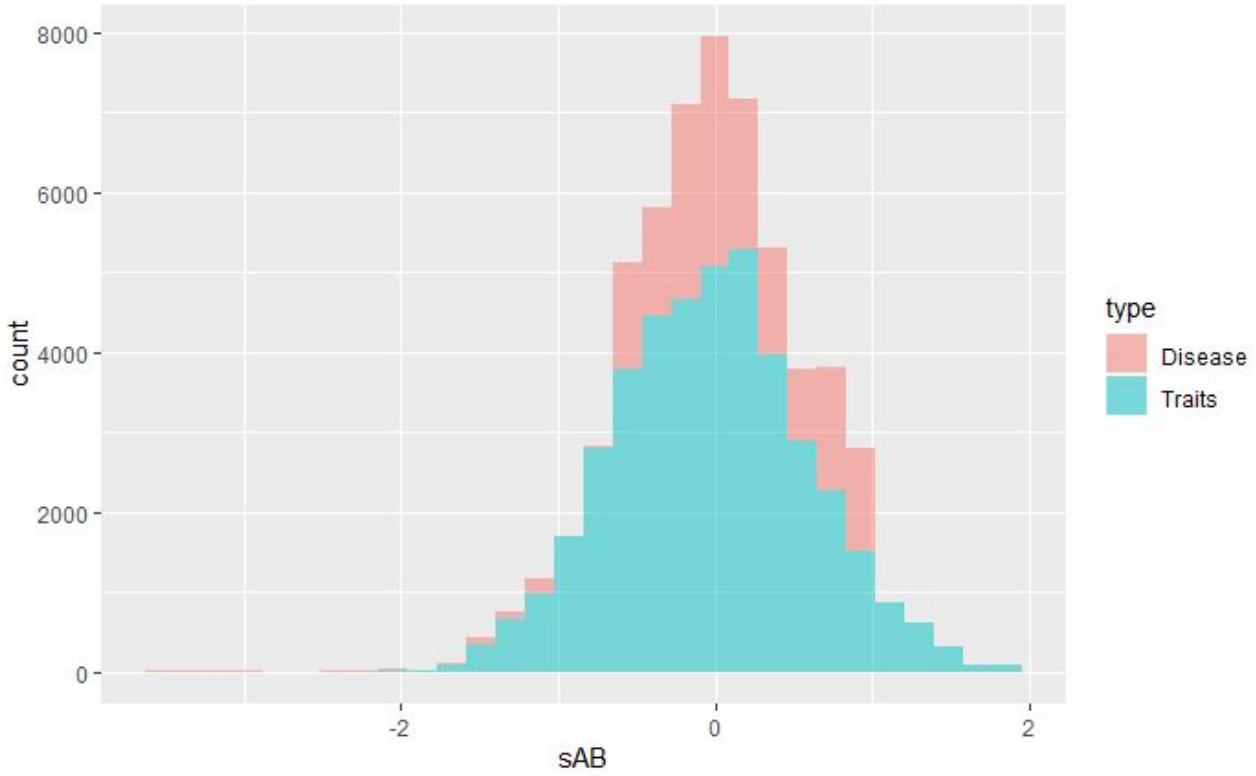

Figure 5: Histogram of sAB values of disease pair associations

The red graph denotes the *sAB* values for disease pairs. Here we see that most of our values lie between -0.5 and 0.5. We also see a significant number of overlapping disease neighborhoods, i.e. it has values less than 0 in the disease pair associations, with most of the values around the 0 mark. On the other hand, we also see that none of the values are greater than 1.

The blue graph denotes the *sAB* values for trait pairs. And what we observe here is similar to a normal distribution of data points which is a contrast to our disease pair associations. Here, we see values ranging from -2 to 2 with a nearly equal number of positive and negative values. However, we also see a significant number of values that go well beyond 1 and even touching 2. This means that these trait modules are topologically separated[1].

6.3 Comparison with the Comorbidity dataset

Park, J et al[2] in their paper in 2009 studied the correlations between the underlying structure of cellular networks and disease comorbidity patterns in the human population. We try to work on one of the datasets used by them in their research to compare our results with the Relative risk values observed by them for a certain set of disease and trait pairs. I have tried to evaluate the *sAB* values with the same disease pairs used in the dataset and observed the results.

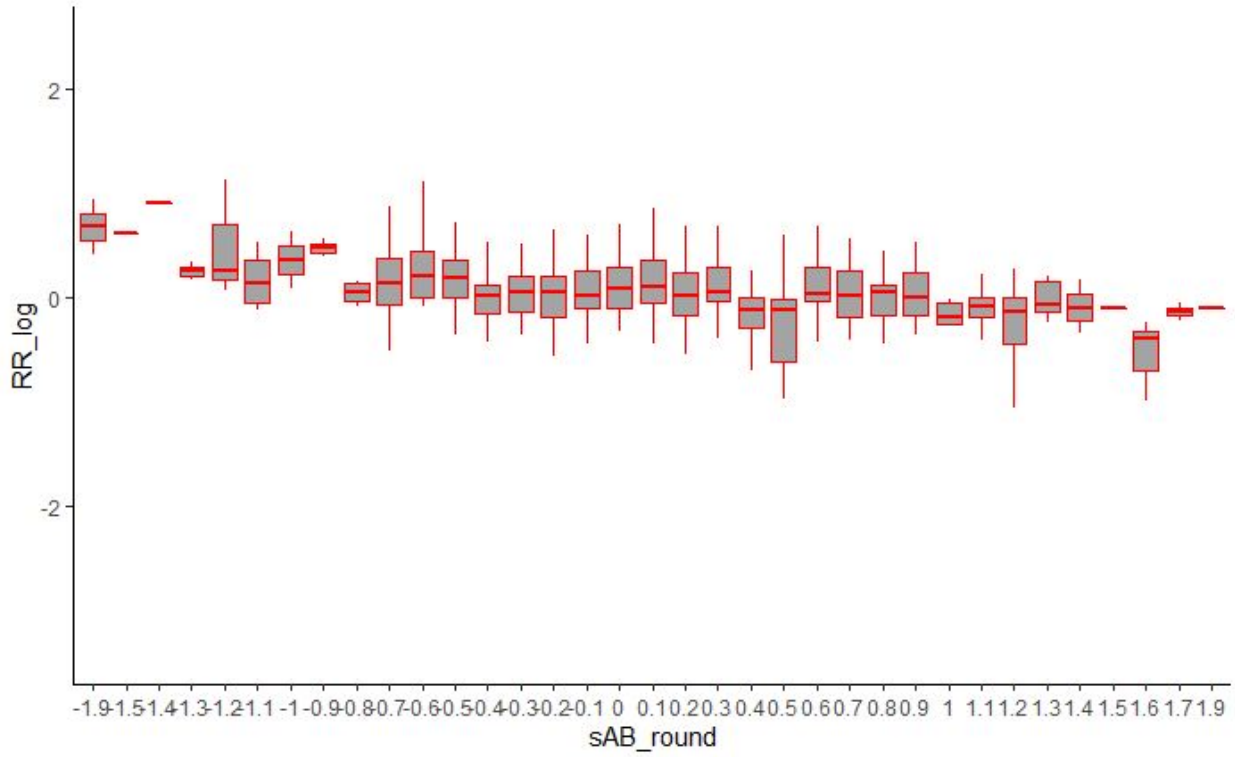

Figure 7: Comorbidity plot

Here we see that Relative Risk (*RR)* values remain nearly constant when the *sAB* values are between -2 to 0.4. Then we see a small decrease in the *RR* value as the *sAB* values increases. Thus, we can say that the lower the *sAB* values, the higher is the relative risk. We will also look at the scatter plot of the *sAB* values against the *RR* values

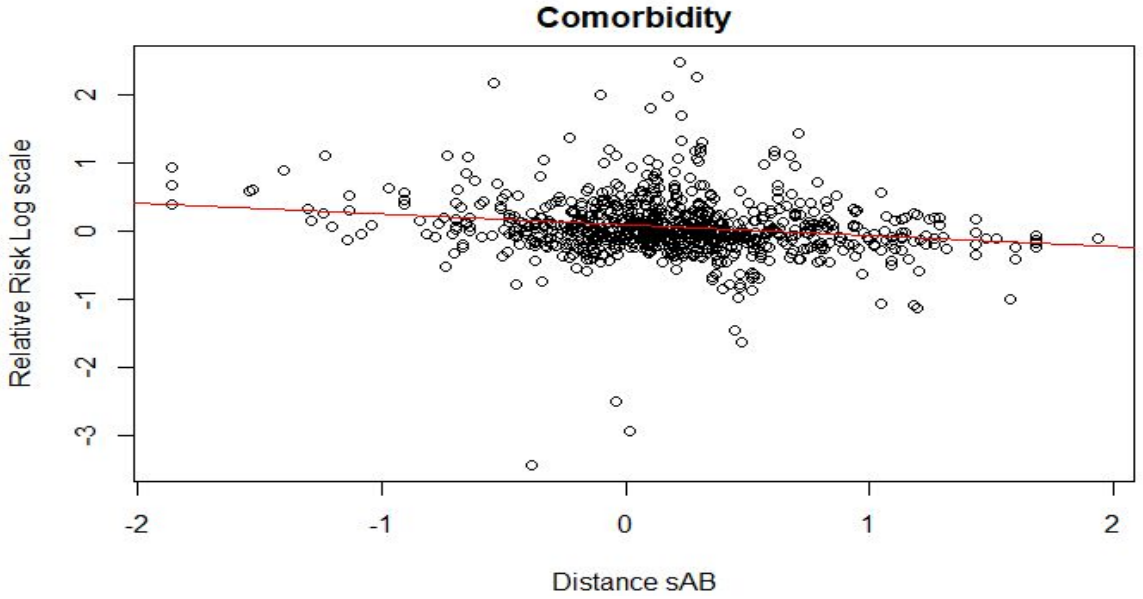

Figure 8: Scatter plot of Comorbidity data and observed values

7. Summary

This project aimed at establishing a model that helps in identifying the relationship between any disease pairs. We were able to pursue a systematic network-based approach to implement a mathematical model defined by *Menche, J. et al.* In this project we established that protein-based positions in a network for two diseases help us understand the biological and clinical similarities between the two diseases. And this can be extended to any disease pairs. We began by developing the network with the *igraph* library in R. Post this, we calculated the individual distances between nodes for disease A ($d_{\overline{AA}}$) and disease B ($d_{\overline{BB}}$) respectively and the distance of the nodes of one disease to the other ($d_{\overline{AB}}$). With these values, we can then calculate the *sAB* values for any disease pairs.

Post this, we observed and compared the *sAB* values obtained in the paper by *Menche, J*, with the values we obtained for similar disease pairs. However, here we obtained a highly uncorrelated graph with data points crowding between 2 points. We then observed the difference in patterns for *sAB* values for disease pairs and trait pairs. Here we observed that the trait pairs graph follows a more normal distribution graph as compared to that of disease pairs. Finally, we also compared the *sAB* values for disease pairs in the comorbidity dataset used by *Park, J, et al*[2] with the Relative Risk values from the same dataset. We observed that as the *sAB* values increases, there is a slight drop in the Relative Risk as well.

8. References

[1] Menche, J., Sharma, A., Kitsak, M., Ghiassian, S. D., Vidal, M., Loscalzo, J., & Barabási, A. L. (2015). Uncovering disease-disease relationships through the incomplete interactome. Science, 347(6224), 1257601.

[2] Park, J., Lee, D. S., Christakis, N. A., & Barabási, A. L. (2009). The impact of cellular networks on disease comorbidity. Molecular systems biology, 5(1).

[3] Rual, J. F., Venkatesan, K., Hao, T., Hirozane-Kishikawa, T., Dricot, A., Li, N., ... & Klitgord, N. (2005). Towards a proteome-scale map of the human protein–protein interaction network. Nature, 437(7062), 1173.

[4] Pinero, J., Queralt-Rosinach, N., Bravo, A., Deu-Pons, J., Bauer-Mehren, A., Baron, M., ... & Furlong, L. I. (2015). DisGeNET: a discovery platform for the dynamical exploration of human diseases and their genes. Database, 2015.

[5] Pyle, D., 1999. Data Preparation for Data Mining. Morgan Kaufmann Publishers, Los Altos, California.

[6] Andreopoulos, W., Labudde, D. (n.d.). Protein-Protein Interaction Networks, Columbia University, New York, USA

[7] R igraph manual pages. (n.d.). Retrieved from https://igraph.org/r/doc/distances.html

[8] Network analysis with R and igraph: NetSci X Tutorial. (2018, July 11). Retrieved from https://kateto.net/networks-r-igraph

9. Appendix

Here, I will present only some of the important code snippets.

9.1 Thresholding Protein interaction links data

```
```{r}
protein[, 3] <- as.numeric(as.character( protein[, 3] ))
protein_links_tresh<- protein[protein[,3]>850, ]
```
```

9.2 Network Creation

```
```{r}
nodes<- unique(protein_links_tresh[,c("protein1")])
nodes<- as.vector(nodes)

network<- graph_from_data_frame(d=protein_links_tresh,
vertices=nodes, directed=T)
network
```
```

9.3 Distance Function Creation

Here, we will highlight the functions created in R that calculates the $d_{\overline{AA}}$, $d_{\overline{BB}}$, and $d_{\overline{AB}}$ values

9.3.1 For Disease A

```
```{r}
disease_A_distance<- function(dA,protein_links_tresh,network){
  dA_vert<- dA %>%
      filter(Protein %in% protein_links_tresh$protein1)

  dA_nodes<- unique(dA_vert[,"Protein"])
  dA_nodes<- as.vector(dA_nodes)

  dis_A<- distances(network, v = dA_nodes, to = dA_nodes, mode =
c("out"), weights = NULL, algorithm = c("automatic"))

  dis_A_df<- as.data.frame(dis_A)
```
```

```
  dis_A_df <- do.call(data.frame, lapply(dis_A_df, function(x) {
  replace(x, is.infinite(x), NA)
  })
)

  dis_A<-data.matrix(dis_A_df)

  ut_disA<- upperTriangle(dis_A)

  dist_A<- mean(ut_disA, na.rm=T)

  return(dist_A)
}
```

9.3.2 For Disease B

```
```{r}
disease_B_distance<- function(dB,protein_links_tresh,network) {
  dB_vert<- dB %>%
      filter(Protein %in% protein_links_tresh$protein1)

  dB_nodes<- unique(dB_vert[,"Protein"])
  dB_nodes<- as.vector(dB_nodes)

  dis_B<- distances(network, v = dB_nodes, to = dB_nodes, mode =
c("out"), weights = NULL, algorithm = c("automatic"))

  dis_B_df<- as.data.frame(dis_B)

  dis_B_df <- do.call(data.frame, lapply(dis_B_df, function(x) {
  replace(x, is.infinite(x), NA)
  })
)

  dis_B<-data.matrix(dis_B_df)

  ut_disB<- upperTriangle(dis_B)

  dist_B<- mean(ut_disB, na.rm=T)

  return(dist_B)
}
```
```

9.3.3 For Diseases A-B

```
```{r}
disease_AB_distance<- function(dA,dB,protein_links_tresh,network) {

  dis_AB<- distances(network, v = dA_nodes, to = dB_nodes, mode =
c("out"), weights = NULL, algorithm = c("automatic"))

  dis_AB_df<- as.data.frame(dis_AB)

  dis_AB_df <- do.call(data.frame, lapply(dis_AB_df, function(x) {
  replace(x, is.infinite(x), NA)
  })
)

  dis_AB<-data.matrix(dis_AB_df)


  dist_AB<- mean(dis_AB, na.rm=T)

  return(dist_AB)
}
```
```

9.4 Disease selection and distance calculation

Here, we will look at the general code to observe the *sAB* value for one instance

9.4.1 Disease Selection

```
```{r}
dA<- dmapp[dmapp$diseaseId=="C0007102", ]
dB<- dmapp[dmapp$diseaseId=="C0011265", ]
```
```

9.4.2 Distance Calculation

```
```{r}
distance_A<-disease_A_distance(dA,protein_links_tresh,network)
distance_B<-disease_B_distance(dB,protein_links_tresh,network)
distance_AB<-disease_AB_distance(dA,dB,protein_links_tresh,network)

#Shortest distance (sAB)
```
```

```
sAB<- distance_AB - (distance_A+distance_B)/2
sAB
```

9.5 Loop to find *sAB* values for the Comorbidity data

Here, *CM[ ]* stands for the comorbidity data

```{r}
magic_for(print, silent = TRUE)
for (i in seq_len(length(CM$Disease1))){
  dA<- dmapp[dmapp$diseaseName==as.character(CM[i,1]), ]
  dB<- dmapp[dmapp$diseaseName==as.character(CM[i,2]), ]

  distance_A<-disease_A_distance(dA,protein_links_tresh,network)
  distance_B<-disease_B_distance(dB,protein_links_tresh,network)
  distance_AB<-disease_AB_distance(dA,dB,protein_links_tresh,network)

  sAB<- distance_AB - (distance_A+distance_B)/2
  print(sAB)
}
magic_result_as_dataframe()
```

9.6 Loop to find *sAB* values for the disease pairs used in the paper

Here, *Dis_names[ ]* is a list of the diseases used in the paper.

```{r}
magic_for(print, silent = TRUE)
for (i in seq_len(length(Dis_names$Disease_ID))){
  for(j in seq_len(length(Dis_names$Disease_ID)-i)){
    dA<- dmapp[dmapp$diseaseId==as.character(Dis_names[i,3]), ]
    dB<- dmapp[dmapp$diseaseId==as.character(Dis_names[j+i,3]), ]

    distance_A<-disease_A_distance(dA,protein_links_tresh,network)
    distance_B<-disease_B_distance(dB,protein_links_tresh,network)

distance_AB<-disease_AB_distance(dA,dB,protein_links_tresh,network)

    sAB<- distance_AB - (distance_A+distance_B)/2
```

```
    output<-c(as.character(Dis_names[i,2]),
as.character(Dis_names[j+i,2]), sAB)

    print(output)
  }
}
magic_result_as_vector()
```

9.7 *sAB* values for Disease and Trait pairs in our dataset

Here we only show the looping code for *sAB* calculations. We have omitted the preprocessing steps

9.7.1 For Disease pairs

```{r}
diseases<- merge[!(merge$diseaseType=="group"), ]
diseases<-diseases[!(diseases$diseaseType=="phenotype"), ]
```
```{r}
diseases_names<-diseases$diseaseId[duplicated(diseases$diseaseId)]
```
```{r}
disease_id<-unique(diseases_names)
disease_id
```
```{r}
magic_for(print, silent = TRUE)
for (i in seq_len(length(disease_id))){
  for(j in seq_len(length(disease_id)-i)){
    dA<- dmapp[dmapp$diseaseId==as.character(disease_id[i]), ]
    dB<- dmapp[dmapp$diseaseId==as.character(disease_id[j+i]), ]

    distance_A<-disease_A_distance(dA,protein_links_tresh,network)
    distance_B<-disease_B_distance(dB,protein_links_tresh,network)

distance_AB<-disease_AB_distance(dA,dB,protein_links_tresh,network)

    sAB<- distance_AB - (distance_A+distance_B)/2

    #output<-c(as.character(Dis_names[i,2]),
as.character(Dis_names[j+i,2]), sAB)
```

```
    print(sAB)
  }
}
disease_result<- magic_result_as_vector()
```

9.7.2 For Trait pairs

```{r}
phenotype<- merge[!(merge$diseaseType=="group"), ]
phenotype<-phenotype[!(phenotype$diseaseType=="disease"), ]
phenotype_names<-phenotype$diseaseId[duplicated(phenotype$diseaseId)]
phenotype_id<-unique(phenotype_names)
phenotype_id <- sample(phenotype_id,300)
phenotype_id
```

```{r}
magic_for(print, silent = TRUE)
for (i in seq_len(length(phenotype_id))){
  for(j in seq_len(length(phenotype_id)-i)){
    dA<- dmapp[dmapp$diseaseId==as.character(phenotype_id[i]), ]
    dB<- dmapp[dmapp$diseaseId==as.character(phenotype_id[j+i]), ]

    distance_A<-disease_A_distance(dA,protein_links_tresh,network)
    distance_B<-disease_B_distance(dB,protein_links_tresh,network)

distance_AB<-disease_AB_distance(dA,dB,protein_links_tresh,network)

    sAB<- distance_AB - (distance_A+distance_B)/2

    #output<-c(as.character(Dis_names[i,2]),
as.character(Dis_names[j+i,2]), sAB)

    print(sAB)
  }
}
phenotype_result<- magic_result_as_vector()
```