Oncology social network analysis and clinical trial enrolments at Princess Margaret Cancer Centre

Benjamin Smith

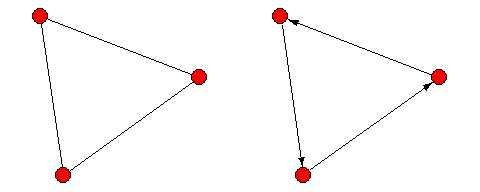
Abstract

Patients at the Princess Margaret Cancer Centre who do not achieve cure or remission following standard treatments often become candidates for clinical trials. Patients who participate in a clinical trial may be suitable for other studies. A key factor influencing further enrollment in subsequent clinical trials is the collaboration network among patient oncologists and most responsible physicians. Possible identification of these collaboration networks can be achieved by analysis of patient movements between intervention types studied in the clinical trials with social network analysis (SNA) and community detection algorithms. The present study evaluates three community detection algorithms: Girvan-Newman, Louvain and an algorithm developed by the author. Girvan-Newman identifies each intervention as their own community, while Louvain groups interventions in a manner that is difficult to interpret. In contrast, the author’s algorithm groups interventions in a way that is both intuitive and informative. This lays the groundwork for future subgroup analysis of clustered interventions.

# Introduction

When cancer patients complete standard treatments at Princess Margaret Cancer Centre and have not responded with being cured or in remission, they become candidates for clinical trials. These clinical trials are regulated studies registered by Health Canada[[1]](#footnote-1) as opposed to quality assurance studies[[2]](#footnote-2). Patients who qualify may be screen failures for other trials, have experienced progressive disease or are receiving maintenance therapy and may be referred to a clinical trial by their oncologist or most responsible physician. Ground truth shows that collaboration networks between oncologists is a primary factor for further engagement in subsequent clinical trials by patients after completion of the given clinical trial they are enrolled in. A possible approach to understanding the structure of these collaboration networks is through use of social network analysis (SNA) and community detection algorithms.

Social network analysis examines individual entities and their relationships among them. The data is represented as a “graph” where individual entities are referred to as “nodes” and their relationships between them as “edges”, which may be directional if specified (see Figure 1). A primary area of study in SNA is the analysis of interconnectivity of nodes, called “communities” and identification of clusters through the use of algorithms called “community detection algorithms”. Rostami et al1 (2023) note that there is no specific model which describes exactly a “community” is. Generally, community detection algorithms employ specific optimization strategies to partition a large-scale complex network into a set of disjoint and compact subgroups, often (but not always) without prior knowledge regarding the number of subgroups and their sizes. Rostami et al further note that it is commonly acknowledged that there is no unique community detection algorithm that can accommodate all kinds of graphs because of the inherent variability in network structures and their respective objective(s).



**Figure 1.** Two simple graphs with directed and undirected edges. Direction is noted by arrowheads at the end of the edges.

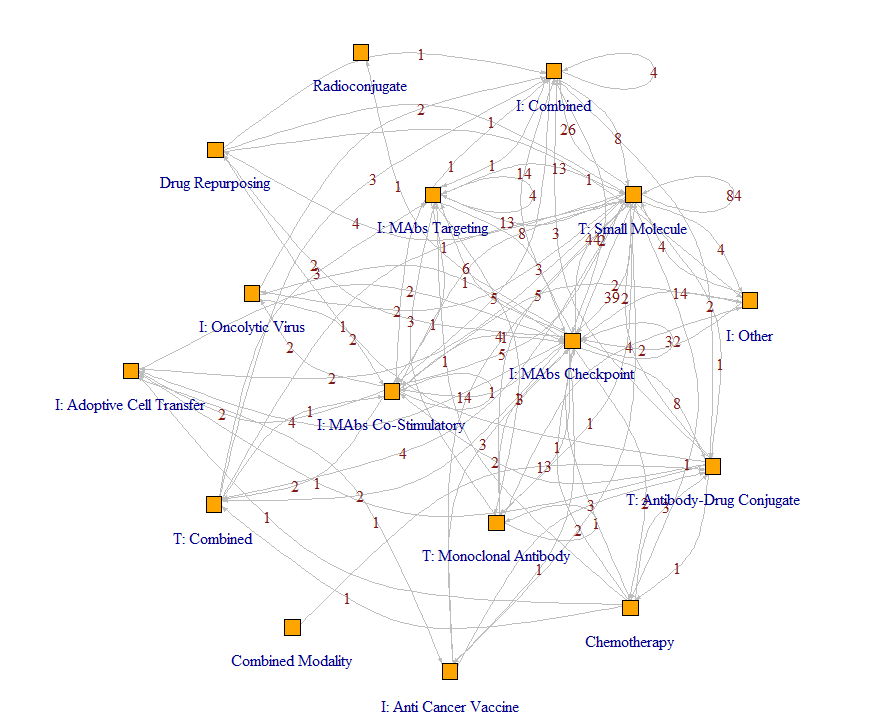
Application of community detection algorithms with oncology clinical trial data has been preformed in the past. Georgiev et al2 (2011) applied the Girvan-Newman3 (2002) algorithm and noted lack of cohesion among researchers who studied treatments for Multiple Myelnoma. Haq and Wang4 (2016) applied the Louvain algorithm (by Blondel et al5 (2008)) to identify communities of cancer patients with significantly different survival curves. The present study applies SNA and compares multiple community detection algorithms to identify collaboration networks between oncologists through the interventions studied in clinical trials via enrollment data of patients in multiple, non-overlapping clinical trials at University Health Network’s Princess Margaret Cancer Centre (PM). Inspired by work from Gorgiev et al (2011), Haq and Wang (2016), Ostovari and Yu6 (2019) and Bissoyi and Patra7 (2020) this research considers the Girvan-Newman and Louvain algorithms and compares them to an author-developed algorithm, referred to as “Smith-Pittman”[[3]](#footnote-3), to identify collaboration networks between clinical trials classified by intervention.

# Materials and Methods

## The Data

The data is a result of multiple data source integration which was undertaken from the Cancer Registry and PM Clinical Research Record and is anonymized. The data spans patient enrollments in oncology clinical trials between January 1st 2016 and December 31st 2018. In this time span, there were 2970 patients enrolled in 515 clinical trials involving 41 principal investigators. For identification of collaboration networks between oncologists, the analytic sample only consists of patients who have been enrolled in more than one clinical trial over within the time span the data captures. The resulting sample consists of 389 patients enrolled in 288 clinical trials. Among the clinical trials, some interventions can be classified into broader categories consisting of Targeted therapies, Immunotherapy, this has been identified in the data with “T:” and “I:” prefixes respectively. The clinical trials are classified by intervention type resulting, in 16 intervention types and 470 patient enrollments. With this classification, the patient referral graph is constructed (see Figure 2).

The analysis is preformed with the R programming language and makes use of an extensive array of libraries and dependencies. The primary libraries that are used are igraph, tidyverse, and tidygraph. For the complete script, refer to see Appendix - Program Syntax.



**Figure 2.** Patient movement between clinical trials classified by intervention type at PM. Nodes indicate the treatment type and the labeled edges indicate movement of patients from trials in a given intervention to another or to the same type of intervention. Among the clinical trials, some interventions can be classified into broader categories consisting of Targeted therapies or Immunotherapy. This has been identified in the data with “T:” and “I:” prefixes respectively.

## Methods

The goal of using community detection algorithms with this data is to identify working groups among interventions based on the movement (incoming and outgoing referrals) of patients between them. These movements in the network are understood through the measures that are considered by the community detection algorithms’ optimization strategies. While the Girvan-Newman, Louvain and Smith-Pittman algorithms differ in their approaches to identification of collaboration networks, their identification strategies are based on maximization of modularity, - a measure that scores the degree of segregation within a network through tightly connected communities or clusters (See Newman8 (2006)).

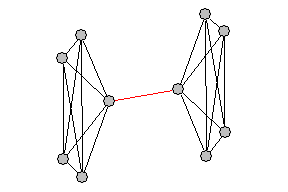
The mathematical representation of modularity is defined in the igraph R package9 (2006) as:

Where is the number of edges (patient movements), is the number of connections shared by nodes and (movements between interventions and ), and are the number of edges coming out from node and going into node (patient movements from intervention and ) and is an indicator variable identifying if nodes and are connected- either directly or through another node (if there is a patient movement between interventions and either directly or through some other intervention). For directed graphs, and are simply the number of connected edges possessed by nodes and respectfully. For a more comprehensive overview modularity and other measures in social network analysis, see Newman (2006), Wasserman and Faust10 (1994) and Latora et al11 (2017).

### Girvan-Newman

The Girvan-Newman algorithm is based on evaluation of the edges of a social network through edge-betweenness centrality. Edge-betweenness centrality is defined by Girvan and Newman (2002) as the number of the shortest paths that go through an edge in a graph divided by the total number of shortest paths between nodes and . Each edge in the graph associated with has their own edge-betweenness centrality value. The igraph (2006) documentation defines edge-betweenness centrality for edge in a social network in mathematical terms as:

Where is the number of shortest paths between nodes and (patient movements between interventions and , either directly or through some other intervention(s)), and is the number of shortest paths which pass through the edge . Figure 3 provides an illustration of a simple network and the edge with the highest edge-betweenness centrality highlighted in red.



**Figure 3.** A simple network demonstrating an edge with a high edge-betweenness centrality, highlighted in red. The network consists of two densely connected clusters, with the red edge serving as the sole connection between them. This edge is crucial for communication between the two clusters, as most of the shortest paths that connect nodes from opposite clusters pass through it.

Edge betweenness can be calculated for directed and undirected edges. As a result, the Girvan-Newman algorithm can be applied to directed or undirected graphs without any transformations. The steps the Girvan-Newman algorithm follows are:

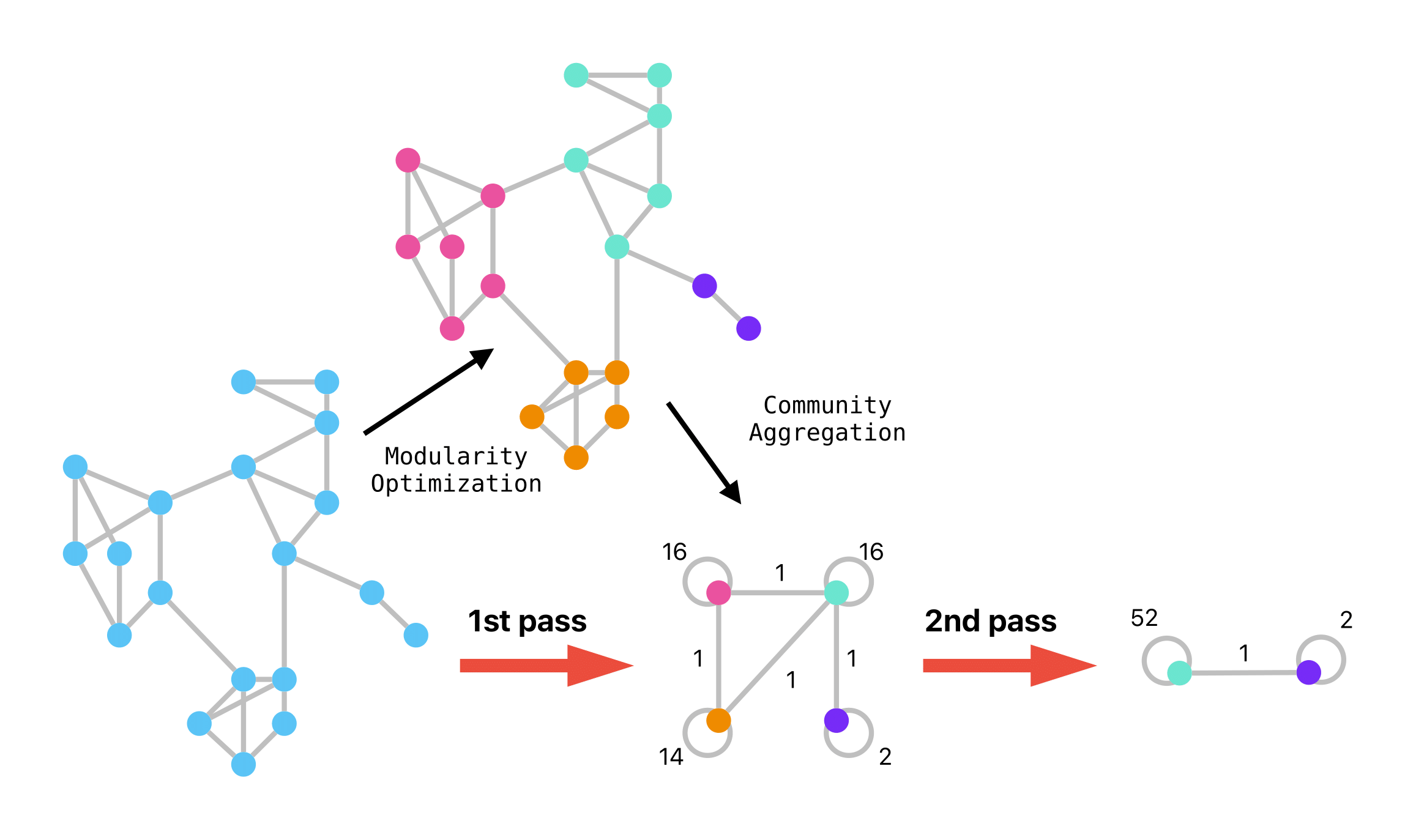
1. Calculate edge-betweenness centrality for all edges in the network.
2. Find the edge with the highest edge-betweenness centrality and remove it from the network.
3. Recalculate edge-betweenness centrality for all remaining edges.
4. Repeat from step 2.

Girvan-Newman can be used when community structure is known and will classify nodes into a pre-deterimined number of communities based on the hierarchy produced by the algorithm (see Girvan and Newman 2002). When the community structure is not known, modularity is evaluated at after each iteration of the algorithm and the grouping of nodes into distinct communities is selected via modularity maximization.

### Louvain

The Louvain algorithm (by Blondel et al 2008) operates in two distinct phases. (i) In the first phase, each node in the network is considered as their own community, resulting in the initial partition with as many communities as there are nodes. The algorithm then assesses the potential modularity gain for each node if it were to leave its current community and join the community of node . After evaluating the potential modularity gain across all communities, node is reassigned to the community of node where the modularity increase is maximized. The process is iteratively and sequentially applied for all nodes until no further improvement can be achieved. This first phase stops when a local maximum of modularity is reached, meaning no individual node moves can enhance modularity. (ii) The second phase involves constructing a new network represent the communites identified in the first phase. Links between nodes of the same community are viewed as “self-loops” for the commmunity in the new network. Once this second phase is complete, the first phase of the algorithm can be reapplied. The combination of these two phases is referred to as a “pass”. The algorithm terminates when there is no other local maxima in modularity to be achieved in subsequent passes.

A key limitation of the Louvain algorithm is that it is generally programmed to work only with undirected graphs[[4]](#footnote-4). In order to apply the Louvain algorithm to a directed graph, it must first be converted to an undirected graph. Figure 4 is a reproduction of Blodel et al’s (2008) illustration of the algorithm.



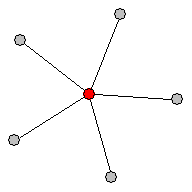
**Figure 4.** Reproduced illustration of the Louvain algorithm (originally designed by Blondel et al (2008)).

### Smith-Pittman

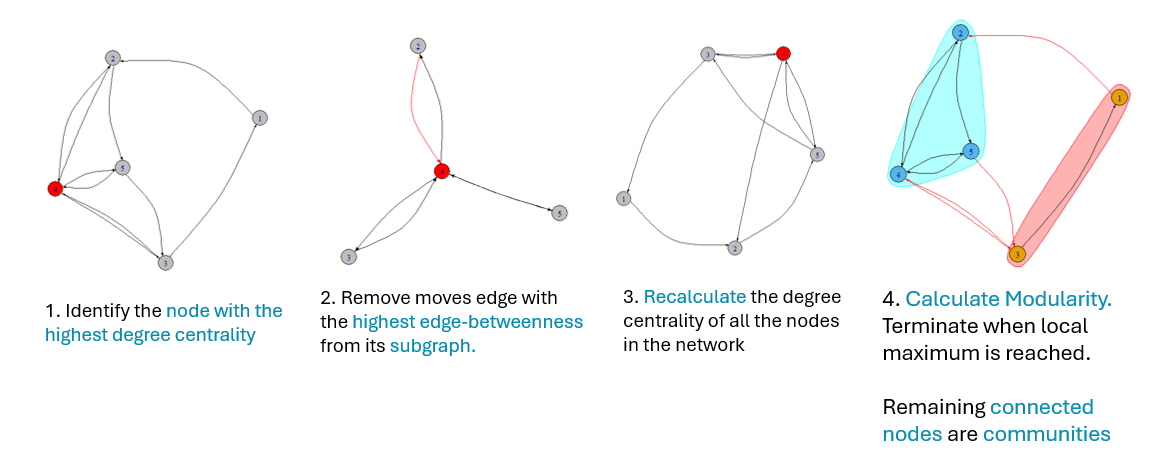
The “Smith-Pittman” algorithm is a modification of the Girvan-Newman algorithm where degree centrality is considered. Degree centrality of a node is simply defined as the number of connections a node has within a given network (see figure 5). The algorithm proceeds through the following steps:

1. Calculate the degree centrality for each node and the edge-betweenness centrality of all edges in the network.
2. Identify the subgraph associated with the node that has the highest degree centrality.
3. Remove the edge possessing the highest calculated edge-betweenness centrality.
4. Recalculate the degree centrality for all nodes and the edge-betweenness centrality for the remaining edges in the network.
5. Repeat from step 2.

Figure 6 provides a visual representation of this algorithm. Like Girvan-Newman, the Smith-Pittman algorithm can be applied both directed and undirected graphs. Conceptually, the algorithm can be specified to terminate once a pre-determined number of communities has been identified. However, its primary design is for use in an unsupervised setting, where clusters are identified through the maximization of modularity as evaluated after each iteration of the algorithm.



**Figure 5.** A simple network highlighting node degree. The center node (colored red) posseses the highest number of connections and as a result posseses the highest degree and degree centrality index.

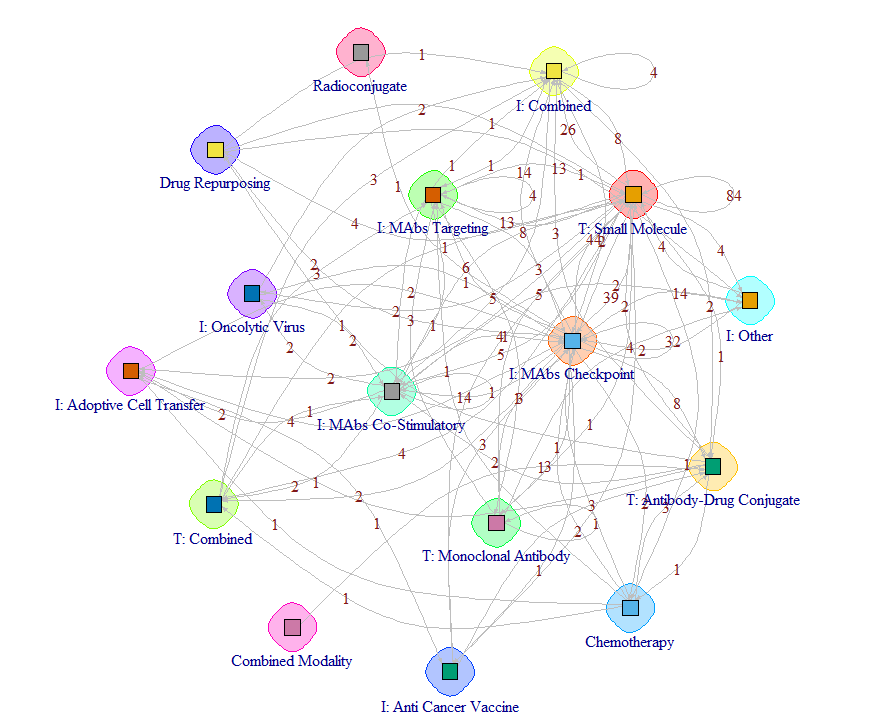


**Figure 6.** Illustration of the Smith-Pittman algorithm. Highlighted convex hulls denote the identification of distinct communities.

# Results

Figures 7-9 show the identified communities by the algorithms by highlighting convex hulls over the grouped interventions. Tables 1-3 shows the grouping of interventions into communities and the breakdown of the number of incoming and outgoing patient referrals for each intervention studied by clinical trials in the PM clinical trial enrollment dataset. Figure 7 demonstrates that the Girvan-Newman algorithm identified each intervention as a separate community ( 0.044) . This result is particularly uninformative, as it is equivalent to not applying any community detection method for collaboration network identification between interventions. Figure 8 shows that the Louvain algorithm groups interventions into four distinct working groups, achieving the highest modularity score ( 0.177). However, the underlying rationale and meaning behind these groupings remains unclear beyond clustering interventions with the objective to maximize modularity.

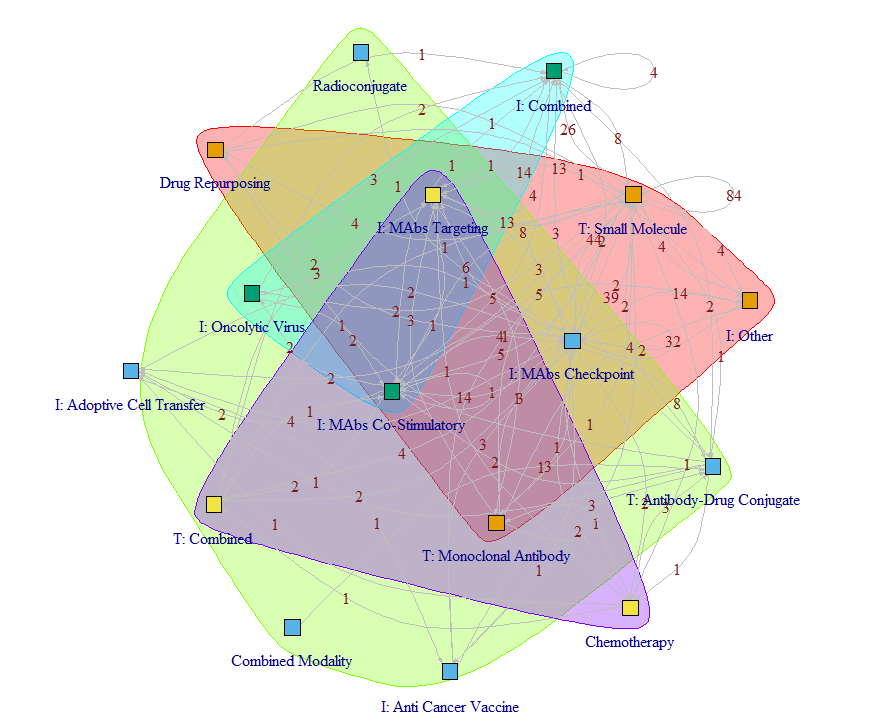
Figure 9 shows that the Smith-Pittman algorithm ( 0.08) identifies eight communities. Six of these communities consist of individual interventions - namely T: Small Molecule, I:MAbs[[5]](#footnote-5) Checkpoint, I:Combined, I:MAbs Targeting, Combined Modality and Radioconjugate - while the remaining two of the communities encompass multiple interventions. The interpretation the communities identified by the Smith-Pittman algorithm can be understood through the degree of connectivity among interventions within these communities. Communities comprising of individual interventions either have the highest or a substantial number of patient referrals, whether incoming from or outgoing to other interventions, or they have the least. Figure 10 illustrates the distribution of interventions by patient referrals, ordered from smallest to largest and highlights the thresholds beyond which single intervention communities are positioned. The interpretation of the communities identified by the Smith-Pittman algorithm suggests that the existence of both highly connected and less connected interventions, as well as broader groups corresponding to typical intervention types. This interpretation offers an intuitive understanding related to the formation of collaboration networks being a function of intervention “popularity” - i.e. patient referrals outgoing and incoming to and from other interventions.



**Figure 7:** Detected communities via Girvan-Newman with modularity maximization. 16 distinct communities.

**Table 1.** Girvan-Newman communities identified. Each intervention is their own community.

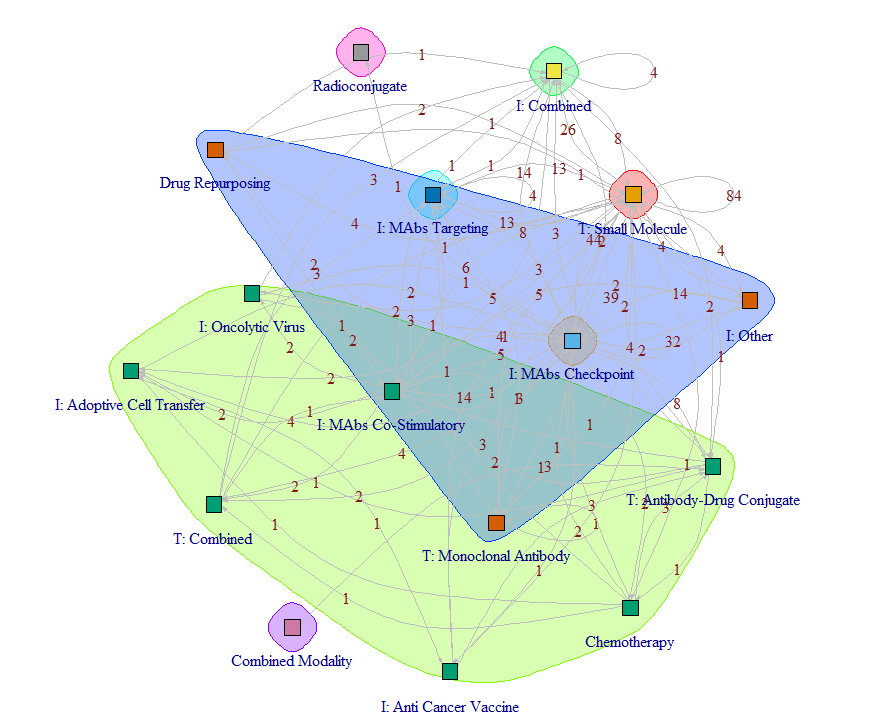
| Intervention | Refferalls In | Referrals Out | Total |
| --- | --- | --- | --- |
| Chemotherapy | 4 | 10 | 14 |
| Combined Modality | 0 | 1 | 1 |
| Drug Repurposing | 7 | 3 | 10 |
| I: Adoptive Cell Transfer | 10 | 3 | 13 |
| I: Anti Cancer Vaccine | 4 | 7 | 11 |
| I: Combined | 54 | 22 | 76 |
| I: MAbs Checkpoint | 92 | 147 | 239 |
| I: MAbs Co-Stimulatory | 31 | 22 | 53 |
| I: MAbs Targeting | 31 | 22 | 53 |
| I: Oncolytic Virus | 4 | 5 | 9 |
| I: Other | 25 | 6 | 31 |
| Radioconjugate | 1 | 0 | 1 |
| T: Antibody-Drug Conjugate | 18 | 10 | 28 |
| T: Combined | 9 | 8 | 17 |
| T: Monoclonal Antibody | 6 | 16 | 22 |
| T: Small Molecule | 174 | 188 | 362 |



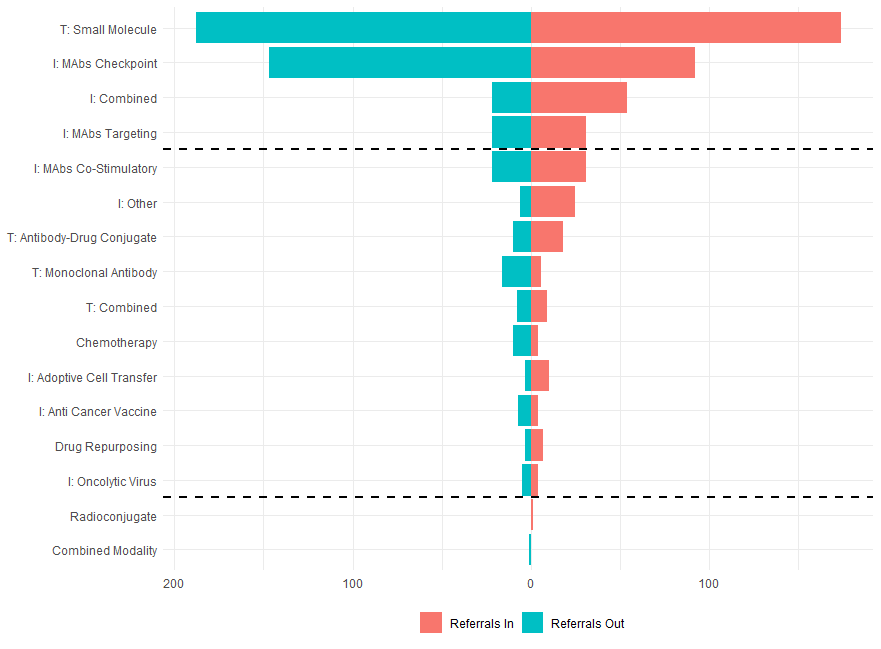
**Figure 8:** Detected communities via Louvain algorithm with modularity maximization. 4 distinct communities.

**Table 2.** Louvain communities identified and their grouped interventions.

| Intervention | Refferalls In | Referrals Out | Total |
| --- | --- | --- | --- |
| Community: 1 | | | |
| Drug Repurposing | 7 | 3 | 10 |
| I: Other | 25 | 6 | 31 |
| T: Monoclonal Antibody | 6 | 16 | 22 |
| T: Small Molecule | 174 | 188 | 362 |
| Community: 2 | | | |
| Combined Modality | 0 | 1 | 1 |
| I: Adoptive Cell Transfer | 10 | 3 | 13 |
| I: Anti Cancer Vaccine | 4 | 7 | 11 |
| I: MAbs Checkpoint | 92 | 147 | 239 |
| Radioconjugate | 1 | 0 | 1 |
| T: Antibody-Drug Conjugate | 18 | 10 | 28 |
| Community: 3 | | | |
| I: Combined | 54 | 22 | 76 |
| I: MAbs Co-Stimulatory | 31 | 22 | 53 |
| I: Oncolytic Virus | 4 | 5 | 9 |
| Community: 4 | | | |
| Chemotherapy | 4 | 10 | 14 |
| I: MAbs Targeting | 31 | 22 | 53 |
| T: Combined | 9 | 8 | 17 |



**Figure 9.** Detected communities via Smith-Pittman algorithm with modularity maximization. 8 distinct communities.



**Figure 10.** Referral distribution among interventions. Interventions outside the boundaries (T: Small Molecule, I:MAbs Checkpoint, I: Combined, I:Mabs Targeting, Radioconjugate and Combined Modality) are each identified as individual communities, while interventions within them are identified as belonging to communities consisting of multiple interventions.

| Intervention | Refferalls In | Referrals Out | Total |
| --- | --- | --- | --- |
| Community: 1 | | | |
| T: Small Molecule | 174 | 188 | 362 |
| Community: 2 | | | |
| I: MAbs Checkpoint | 92 | 147 | 239 |
| Community: 3 | | | |
| Chemotherapy | 4 | 10 | 14 |
| I: Adoptive Cell Transfer | 10 | 3 | 13 |
| I: Anti Cancer Vaccine | 4 | 7 | 11 |
| I: MAbs Co-Stimulatory | 31 | 22 | 53 |
| I: Oncolytic Virus | 4 | 5 | 9 |
| T: Antibody-Drug Conjugate | 18 | 10 | 28 |
| T: Combined | 9 | 8 | 17 |
| Community: 4 | | | |
| I: Combined | 54 | 22 | 76 |
| Community: 5 | | | |
| I: MAbs Targeting | 31 | 22 | 53 |
| Community: 6 | | | |
| Drug Repurposing | 7 | 3 | 10 |
| I: Other | 25 | 6 | 31 |
| T: Monoclonal Antibody | 6 | 16 | 22 |
| Community: 7 | | | |
| Combined Modality | 0 | 1 | 1 |
| Community: 8 | | | |
| Radioconjugate | 1 | 0 | 1 |

**Table 3.** Smith-Pittman communities identified and their grouped interventions.

# Discussion

Where the Girvan-Newman algorithm failed to identify communities, the Louvain and Smith-Pittman algorithms succeeded. A possible explanation for this discrepancy lies in the nature of the data analyzed, which includes patient referrals to clinical trials that investigate the same intervention types as the clinical trials patients were previously enrolled in. In graph theory, such referrals are represented as “self-loops” and introduce complexity in the network. The Girvan-Newman algorithm- whose original design was not for complex networks - struggles in such contexts, leading to its failure to group multiple interventions into communities based on modularity maximization.

The Louvain algorithm successfully detected communities, however, the resulting groups were difficult to interpret. This difficulty arises because the Louvain algorithm bases its community selection purely on modularity maximization and does not consider the direction of patient movements the underlying structural or functional significance of particular interventions in the context of the network. The primary advantage of the Louvain algorithm is its efficency in preforming community detection on large networks. It has been widely used in applications such as the Twitter Social Network (Pujol et al. 2009) which consisted of 2.4 million nodes and 38 million links and mobile phone network data (Greene et al. 2010) with 4 million nodes, 100 million links. These networks are orders of magnitude larger than the patient referral network analyzed in this study, highlighting the scalability of the Louvain algorithm. However, utility of such a algorithm is limited in smaller, more specialized networks where interpretability and justification of communities identified is important.

Contrasting, the Smith-Pittman algorithm directly addresses connectivity of interventions studied in the clinical trials by incorperating degree centrality and edge-betweenness centrality. This approach allows for the identification of communities with a more ordered structure, distinguishing between highly connected and minor interventions as they reflect the relational dynamics in the network. The results from the Smith-Pittman algorithm are promising, however the results from this analysis alone is insufficient to establish generalizability of the algorithm. To fully assess its usefulness, a formal simulation study and further application of the Smith-Pittman algorithm in diverse settings is necessary. Additionally, the practical value of identified communities will become evident when they are applied as grouping variables in downstream analysis, such as outcome prediction or intervention effectiveness studies.

Further research should focus on subgroup analysis and exploring extensions back to traditional statistical methods, such as regression and survival analysis. This research can further validate the utility of the identified communities and use of SNA and community detection algorithms in clinical research settings. The results of the Smith-Pittman algorithm lay the groundwork for these efforts and potentially offer a robust tool for community detection in social and complex networks. Further work with the identified communities should involve assessment of the impact of community structure on patient outcomes and identify if there are any structural inequities present in the clinical trial enrollments. This line of research can lead to the identification of collaboration networks that improve patient care in clinical settings.

# References

1. Rostami, M., Oussalah, M., Berahmand, K. & Farrahi, V. Community Detection Algorithms in Healthcare Applications: A Systematic Review. IEEE Access 11, 30247–30272 (2023).

2. Georgiev, H., Tsalatsanis, A., Kumar, A. & Djulbegovic, B. Social Network Analysis (SNA) of Research Programs In Multiple Myeloma (MM). Blood 118, 3144–3144 (2011).

3. Girvan, M. & Newman, M. E. J. Community structure in social and biological networks. Proceedings of the National Academy of Sciences 99, 7821–7826 (2002).

4. Haq, N. & Wang, Z. J. Community detection from genomic datasets across human cancers. in 2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP) 1147–1150 (IEEE, 2016). doi:10.1109/GlobalSIP.2016.7906021.

5. Blondel, V. D., Guillaume, J.-L., Lambiotte, R. & Lefebvre, E. Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment 2008, P10008 (2008).

6. Ostovari, M. & Yu, D. Impact of care provider network characteristics on patient outcomes: Usage of social network analysis and a multi-scale community detection. PLoS One 14, e0222016 (2019).

7. Bissoyi, S. & Patra, M. R. Community Detection in a Patient-Centric Social Network. in 171–182 (2021). doi:10.1007/978-981-15-7394-1\_17.

8. Newman, M. E. J. Modularity and community structure in networks. Proceedings of the National Academy of Sciences 103, 8577–8582 (2006).

9. Csardi, G. & Nepusz, T. The igraph software package for complex network research. InterJournal, Complex Systems 1695 (2006).

10. Wasserman, S. & Faust, K. Social Network Analysis. (Cambridge University Press, 1994). doi:10.1017/CBO9780511815478.

11. Latora, V., Nicosia, V. & Russo, G. Complex Networks. (Cambridge University Press, 2017). doi:10.1017/9781316216002.

12. Dugué, N. & Perez, A. Direction matters in complex networks: A theoretical and applied study for greedy modularity optimization. Physica A: Statistical Mechanics and its Applications 603, 127798 (2022).

# Appendix - Program Syntax

# Pre-Processing Script  
library(tidyverse)  
library(tidygraph)  
library(igraph)  
library(ig.degree.betweenness) # Author developed methodology, pending public release  
library(plyr) # for join\_all  
library(gt) # for tables  
# Load R Data  
real\_df <- readRDS("C:/Users/ben29/OneDrive - University of Toronto/UofT/Fall2023/CHL5208/UHN/DLSPH\_ClinT\_subjects\_AEs\_2016\_2018\_19Dec2023.rds")  
  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Immunotherapy- MAbs-immunomodulatory-Checkpoint")] <- "I: MAbs Checkpoint";  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Tageted therapy- antibody-drug conjugate")] <- "T: Antibody-Drug Conjugate";  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Immunotherapy- MAbs-immunomodulatory-Co-Stimulatory")] <- "I: MAbs Co-Stimulatory";  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Immunotherapy- Immuno + other investigational agent")] <- "I: Combined";  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Targeted therapy - combined (small molecule + monoclonal antibody)")] <- "T: Combined";  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Immunotherapy- MAbs- Tumour-targeting (includes immunoconjugates, naked MAbs)")] <- "I: MAbs Targeting";  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Targeted therapy - small molecule")] <- "T: Small Molecule";  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Immunotherapy- Other")] <- "I: Other";  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Targeted therapy - monoclonal antibody")] <- "T: Monoclonal Antibody";  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Immunotherapy- Adoptive Cell Transfer (e.g. TILS)")] <- "I: Adoptive Cell Transfer";  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Immunotherapy- combined types")] <- "I: Combined";  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Other - drug repurposing")] <- "Drug Repurposing";  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Immunotherapy- Cytokines (eg. INFa, IL, Hematopoietic growth factors)")] <- "I: MAbs Co-Stimulatory";  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Multiple- Biomarker Targeted")] <- "T: Combined";  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Immunotherapy- Anti Cancer Vaccine- Peptide based vaccine")] <- "I: Anti Cancer Vaccine";  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Chemotherapy")] <- "Chemotherapy";  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Immunotherapy- Oncolytic Virus")] <- "I: Oncolytic Virus";  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Combined modality (e.g chemoradiation, EBRT+Brachy)")] <- "Combined Modality";  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Immunotherapy- Anti Cancer Vaccine- Gene Therapy (e.g DNA/RNA vaccines)")] <- "I: Anti Cancer Vaccine";  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Other - radioconjugate")] <- "Radioconjugate";  
real\_df$New\_Intervention\_Name[real\_df$New\_Intervention\_Name %in% c("Homonal Treatment")] <- "Drug Repurposing";  
  
  
intervention\_graph\_real\_directed<- real\_df |>  
 dplyr::group\_by(Subject\_ID,Study\_ID) |>  
 dplyr::filter(dplyr::n() > 1) |>  
 dplyr::distinct(pick(Subject\_ID,Study\_ID),.keep\_all = TRUE) |>  
 dplyr::ungroup() |>  
 dplyr::filter(Subject\_ID %in% names(table(Subject\_ID))[table(Subject\_ID) > 1]) |>  
 dplyr::group\_by(Subject\_ID) |>  
 dplyr::group\_split() |>  
 lapply(function(x) x |>  
 dplyr::mutate(x, index = 1:nrow(x),  
 direction = ifelse(index%%2 == 1, "from","to"))) |>  
 do.call(what = rbind) |>  
 dplyr::select(Subject\_ID, Study\_ID, direction, New\_Intervention\_Name) |>  
 tidyr::pivot\_wider(  
 id\_cols = c(Subject\_ID),  
 names\_from = direction,  
 values\_from = c(New\_Intervention\_Name, Study\_ID)) |>  
 dplyr::rename(from = New\_Intervention\_Name\_from,  
 to = New\_Intervention\_Name\_to,  
 Study\_ID = Study\_ID\_from) |>  
 tidyr::unnest(from) |>  
 tidyr::unnest(to) |>  
 tidyr::unnest(Study\_ID) |>  
 tidyr::unnest(Study\_ID\_to) |>  
 dplyr::mutate(from = str\_wrap(from, width = 30),  
 to = str\_wrap(to, width = 30)) |>  
 #dplyr::group\_by(from, to) |>  
 #dplyr::count(name="Num\_Patients") |>  
 tidygraph::as\_tbl\_graph(directed = TRUE) |>  
 igraph::as.igraph()  
  
  
intervention\_graph\_real\_undirected<- real\_df |>  
 dplyr::group\_by(Subject\_ID) |>  
 dplyr::filter(dplyr::n() > 1) |>  
 dplyr::distinct(pick(Subject\_ID,Study\_ID),.keep\_all = TRUE) |>  
 dplyr::ungroup() |>  
 dplyr::filter(Subject\_ID %in% names(table(Subject\_ID))[table(Subject\_ID) > 1]) |>  
 dplyr::group\_by(Subject\_ID) |>  
 dplyr::group\_split() |>  
 lapply(function(x) x |>  
 dplyr::mutate(x, index = 1:nrow(x),  
 direction = ifelse(index%%2 == 1, "from","to"))) |>  
 do.call(what = rbind) |>  
 dplyr::select(Subject\_ID, Study\_ID, direction, New\_Intervention\_Name) |>  
 tidyr::pivot\_wider(  
 id\_cols = c(Subject\_ID),  
 names\_from = direction,  
 values\_from = c(New\_Intervention\_Name, Study\_ID)) |>  
 dplyr::rename(from = New\_Intervention\_Name\_from,  
 to = New\_Intervention\_Name\_to,  
 Study\_ID = Study\_ID\_from) |>  
 tidyr::unnest(from) |>  
 tidyr::unnest(to) |>  
 tidyr::unnest(Study\_ID) |>  
 tidyr::unnest(Study\_ID\_to) |>  
 dplyr::mutate(from = str\_wrap(from, width = 30),  
 to = str\_wrap(to, width = 30)) |>  
 #dplyr::group\_by(from, to) |>  
 #dplyr::count(name="Num\_Patients") |>  
 tidygraph::as\_tbl\_graph(directed = FALSE) |>  
 igraph::as.igraph()  
  
  
own\_subj\_mult\_studies\_check <- real\_df |>  
 dplyr::distinct(Subject\_ID, Study\_ID) |>  
 dplyr::group\_by(Subject\_ID) |>  
 dplyr::count(name="N\_Studies") |>  
 dplyr::filter(N\_Studies > 1)  
#389 participants enrolled in more than 1 study in 470 instances;  
  
  
  
### Limit analysis to participants who enrolled in more than 1 clinical trial;  
  
  
own <- real\_df |>  
 # Adding this line because Tyler has it as well.   
 dplyr::mutate(eligible = "eligible") |>  
 dplyr::filter(Subject\_ID %in% unique(own\_subj\_mult\_studies\_check$Subject\_ID)) |>  
 dplyr::select(  
 "Subject\_ID",  
 "Study\_ID",  
 "Enrolled\_Date\_Time",  
 "New\_Intervention\_Name",  
 "PI\_ID",  
 "AE\_Grade\_3\_Plus",  
 "New\_Intervention\_Name",  
 "eligible",  
 "Age\_40",  
 "Age\_65",  
 "Baseline\_AE",  
 "New\_Int\_Name",  
 "Phase",  
 "Randomized",  
 "Combination",  
 "Sponsor\_Type",  
 "Disease\_Site\_Group"  
 )  
  
  
  
own\_check <- own |>  
 dplyr::select(Subject\_ID, Study\_ID, New\_Intervention\_Name, PI\_ID) |>  
 dplyr::filter(Subject\_ID %in% unique(own\_subj\_mult\_studies\_check$Subject\_ID)) |>  
 dplyr::distinct(Subject\_ID, Study\_ID, New\_Intervention\_Name, PI\_ID) |>  
 dplyr::group\_by(Subject\_ID, Study\_ID, New\_Intervention\_Name, PI\_ID)  
  
own\_check <- as.data.frame(own\_check)  
  
# length(own\_check[!duplicated(own\_check$Subject\_ID), ]$Subject\_ID); #389 unique subjects;  
# length(own\_check[!duplicated(own\_check$Study\_ID), ]$Study\_ID); #288 unique studies;  
# length(own\_check[!duplicated(own\_check$PI\_ID), ]$PI\_ID); #41 unique PIs;  
# length(own\_check[!duplicated(own\_check$New\_Intervention\_Name), ]$New\_Intervention\_Name); #16 unique interventions;  
# length(unique(own\_check[duplicated(own\_check$Subject\_ID), ]$Subject\_ID)); #389 subjects in in multiple studies;  
# length(own\_check[duplicated(own\_check$Subject\_ID), ]$Subject\_ID); #470 instances of subjects in multiple studies;  
  
###have to do New\_Intervention\_Name in here for correct department;  
linkedDataStudies\_0 <- own |>  
 dplyr::distinct(New\_Intervention\_Name, Study\_ID, Subject\_ID, .keep\_all = TRUE) |>  
 dplyr::group\_by(New\_Intervention\_Name, Study\_ID) |>  
 dplyr::count(name = "Num\_Patients")  
  
  
linkedDataStudies <- own |>  
 dplyr::select(New\_Intervention\_Name) |>  
 dplyr::group\_by(New\_Intervention\_Name)  
  
linkedDataPIs\_0 <- own |>  
 dplyr::distinct(PI\_ID, New\_Intervention\_Name, Study\_ID, Subject\_ID) |>  
 dplyr::group\_by(PI\_ID, New\_Intervention\_Name, Study\_ID) |>  
 dplyr::count(name = "Num\_Patients")  
  
  
linkedDataPIs <- own |>  
 dplyr::distinct(PI\_ID, New\_Intervention\_Name, Study\_ID) |>  
 dplyr::group\_by(PI\_ID)  
  
linkedDataSubject\_ID <- own |>  
 dplyr::select(  
 Subject\_ID,  
 Enrolled\_Date\_Time,  
 Study\_ID,  
 New\_Intervention\_Name,  
 New\_Int\_Name,  
 eligible,  
 Combination,  
 Randomized,  
 AE\_Grade\_3\_Plus,  
 Age\_65  
 ) |>  
 dplyr::arrange(  
 Subject\_ID,  
 Enrolled\_Date\_Time,  
 Study\_ID,  
 New\_Intervention\_Name,  
 New\_Int\_Name,  
 eligible,  
 desc(AE\_Grade\_3\_Plus)  
 ) |>  
 dplyr::group\_by(Subject\_ID, Study\_ID, New\_Intervention\_Name) |>  
 dplyr::filter(row\_number() == 1)  
  
reach2=function(x){  
 r=vector(length=vcount(x))  
 for (i in 1:vcount(x)){  
 n=neighborhood(x,2,nodes=i)  
 ni=unlist(n)  
 l=length(ni)  
 r[i]=(l)/vcount(x)}  
 r}  
  
reach3=function(x){  
 r=vector(length=vcount(x))  
 for (i in 1:vcount(x)){  
 n=neighborhood(x,3,nodes=i)  
 ni=unlist(n)  
 l=length(ni)  
 r[i]=(l)/vcount(x)}  
 r}  
  
dwreach=function(x){  
 distances=shortest.paths(x) #create matrix of geodesic distances  
 diag(distances)=1 # replace the diagonal with 1s  
 weights=1/distances # take the reciprocal of distances  
 apply(weights,1,sum) # sum for each node (row)  
}  
  
#  
  
fpntable <- table(own$Subject\_ID);  
otable <- table(own$PI\_ID);  
rtable <- table(own$New\_Intervention\_Name);  
ownSmall <- own;  
three\_way\_count <- ownSmall |>  
 dplyr::select(PI\_ID, New\_Intervention\_Name, Study\_ID, Subject\_ID) |>  
 dplyr::group\_by(PI\_ID, New\_Intervention\_Name, Study\_ID, Subject\_ID) |>  
 dplyr::count(name="freq")  
three\_way\_count <- as.data.frame(three\_way\_count);  
  
  
edgelist <- cbind(three\_way\_count$PI\_ID, three\_way\_count$New\_Intervention\_Name, three\_way\_count$Study\_ID, three\_way\_count$Subject\_ID, three\_way\_count$freq);  
  
colnames(edgelist) <- c("PI\_ID", "New\_Intervention\_Name", "Study\_ID", "Subject\_ID", "freq");  
edgelist <- as.data.frame(edgelist);  
linkedDataPIs\_0 <- as.data.frame(linkedDataPIs\_0);  
linkedDataPIs <- as.data.frame(linkedDataPIs);  
linkedDataStudies\_0 <- as.data.frame(linkedDataStudies\_0);  
linkedDataStudies <- as.data.frame(linkedDataStudies);  
linkedDataSubject\_ID <- as.data.frame(linkedDataSubject\_ID);  
  
edgelist0 <- join\_all(list(edgelist, linkedDataPIs\_0, linkedDataStudies\_0), by = c("New\_Intervention\_Name", "Study\_ID"), type = "left", match = "first");  
edgelist00 <- join\_all(list(edgelist0, linkedDataSubject\_ID), by=c("Subject\_ID", "New\_Intervention\_Name", "Study\_ID"), type="left", match = "first");  
edgelist00\_tibble <- as\_tibble(edgelist00);  
  
  
edgelist <- edgelist00[,colnames(edgelist00) %in% c("Subject\_ID", "Study\_ID", "Enrolled\_Date\_Time", "New\_Intervention\_Name", "PI\_ID", "Num\_Patients", "eligible", "Randomized", "Combination", "freq", "Department", "Enrolled\_Date\_Time", "Status\_Change\_Date\_Time", "New\_Int\_Name", "AE\_Grade\_3\_Plus", "Age\_65")];  
  
edgelist\_count <- length(edgelist$Subject\_ID);  
  
n\_pi <- length(unique(edgelist$PI\_ID));  
n\_subjects <- length(unique(edgelist$Subject\_ID));  
n\_studies <- length(unique(edgelist$New\_Intervention\_Name));  
strat <- unique(eval(parse(text=paste("edgelist$", "eligible", sep=""))));  
strat <- na.omit(strat);  
  
edgelist <- edgelist[order(edgelist$Subject\_ID, edgelist$Enrolled\_Date\_Time, edgelist$New\_Intervention\_Name),];  
#which(is.na(eval(parse(text=paste("edgelist$", var[k], sep=""))))); #none, good check;  
  
edgelist <- edgelist[,colnames(edgelist) %in% c("Subject\_ID", "Study\_ID", "New\_Intervention\_Name", "PI\_ID", "Num\_Patients", "freq", "eligible", "Randomized", "Combination", "Department", "Enrolled\_Date\_Time", "Status\_Change\_Date\_Time", "New\_Int\_Name", "AE\_Grade\_3\_Plus", "Age\_65")];  
  
edgelist$Subject\_ID <- as.character(edgelist$Subject\_ID);  
edgelist$Study\_ID <- as.character(edgelist$Study\_ID);  
edgelist$PI\_ID <- as.character(edgelist$PI\_ID);  
edgelist$New\_Intervention\_Name <- as.character(edgelist$New\_Intervention\_Name);  
edgelist$freq <- as.numeric(as.character(edgelist$freq));  
  
edgelistPre\_st <- edgelist  
  
counterStrat <- length(strat)  
  
st = strat;  
strataCat = st;  
edgelist <- edgelistPre\_st |>  
 filter(eval(parse(text="eligible")) == st)  
edgelist <- as.data.frame(edgelist);  
n\_studies\_strata <- length(unique(edgelist$New\_Intervention\_Name));  
  
edgelist <- edgelist[order(edgelist$Subject\_ID, edgelist$Enrolled\_Date\_Time, edgelist$New\_Intervention\_Name),];  
  
  
###  
### DO THIS FOR A SIMPLER DATASET TO CHECK CODE;  
###  
edgelist <- edgelist |>  
 dplyr::arrange(Subject\_ID, Enrolled\_Date\_Time) |>  
 dplyr::group\_by(Subject\_ID) |>  
 dplyr::mutate(order = row\_number()) |>  
 dplyr::mutate(from = Study\_ID,   
 to = Study\_ID,  
 order\_from = order,   
 order\_to = order)  
  
igraph0 <- edgelist |>  
 dplyr::bind\_rows(edgelist) |>  
 dplyr::arrange(Subject\_ID, Study\_ID) |>  
 dplyr::group\_by(Subject\_ID,Study\_ID) |>  
 dplyr::ungroup() |>  
 dplyr::group\_by(Subject\_ID) |>  
 dplyr::group\_split() |>  
 lapply(function(x) x |>  
 dplyr::mutate(x, index = 1:nrow(x),  
 direction = ifelse(index%%2 == 1, "from","to"))) |>  
 do.call(what = rbind) |>  
 dplyr::select(Subject\_ID, Study\_ID, direction) |>  
 tidyr::pivot\_wider(  
 id\_cols = c(Subject\_ID),  
 names\_from = direction,  
 values\_from = c(Study\_ID)) |>  
 tidyr::unnest(from, .drop=TRUE) |>  
 tidyr::unnest(to, .drop=TRUE)   
  
igraph1 <- plyr::join\_all(list(igraph0, edgelist[,c("Subject\_ID", "from",  
 "order\_from")]), by=c("Subject\_ID", "from"), type='left');  
igraph2 <- plyr::join\_all(list(igraph1, edgelist[,c("Subject\_ID", "to",  
 "order\_to")]), by=c("Subject\_ID", "to"), type='left');  
  
igraph2 <- igraph2 |>  
 dplyr::filter(order\_from < order\_to) |>  
 dplyr::arrange(Subject\_ID, order\_from, order\_to) |>  
 dplyr::group\_by(Subject\_ID, from) |>  
 dplyr::filter(row\_number() == 1) |>  
 dplyr::mutate(Study\_ID\_from = from,  
 Study\_ID\_to = to)   
  
edgelist <- edgelist |>  
 dplyr::arrange(Subject\_ID, Enrolled\_Date\_Time) |>  
 dplyr::group\_by(Subject\_ID) |>  
 dplyr::mutate(order = row\_number()) |>  
 dplyr::mutate(New\_Intervention\_Name\_from = New\_Intervention\_Name,   
 New\_Intervention\_Name\_to = New\_Intervention\_Name)  
  
igraph3 <- plyr::join\_all(list(igraph2, edgelist[,c("Subject\_ID", "from",   
 "New\_Intervention\_Name\_from")]), by=c("Subject\_ID", "from"), type='left');  
igraph4 <- plyr::join\_all(list(igraph3, edgelist[,c("Subject\_ID", "to",   
 "New\_Intervention\_Name\_to")]), by=c("Subject\_ID", "to"), type='left');  
igraph5 <- igraph4 |>  
 dplyr::mutate(Study\_ID\_from = from,  
 Study\_ID = to,  
 from = New\_Intervention\_Name\_from,  
 to = New\_Intervention\_Name\_to) |>  
 dplyr::select(-c("New\_Intervention\_Name\_from", "New\_Intervention\_Name\_to"))  
  
igraph <- igraph5 |>  
 dplyr::mutate(from = str\_wrap(from, width = 30),  
 to = str\_wrap(to, width = 30)) |>  
 tidygraph::as\_tbl\_graph(directed = TRUE) |>  
 igraph::as.igraph()  
  
e <- igraph::get.edgelist(igraph, names=FALSE);  
l <- qgraph::qgraph.layout.fruchtermanreingold(e, vcount=vcount(igraph), area=30\*(vcount(igraph)^2),repulse.rad=(vcount(igraph)^2.1));  
  
# ########## Do this for a simpler graph just before plotting;  
igraph\_simplified <- igraph  
E(igraph\_simplified)$weight <- 1  
igraph\_simplified <- igraph::simplify(  
 igraph\_simplified,  
 remove.multiple = T,  
 remove.loops = F,  
 edge.attr.comb = list(weight = "sum", "ignore")  
)  
E(igraph\_simplified)$label <- E(igraph\_simplified)$weight  
  
  
# FOR VISUALS IN THIS REPORT  
  
# Figure 1  
  
g\_directed <- graph(c(1, 2, 2, 3, 3, 1), directed = TRUE)  
g\_undirected <- as.undirected(g\_directed)  
  
V(g\_directed)$color <- "red"  
V(g\_undirected)$color <- "red"  
E(g\_directed)$color <- "black"  
E(g\_undirected)$color <- "black"  
set.seed(5208)  
par(mfrow= c(1,2),mar=c(0,0,0,0)+.1)  
plot(g\_undirected,  
 vertex.label = "",  
 edge.arrow.size = 0.5,  
 vertex.size = 20)  
set.seed(5208)  
plot(g\_directed,  
 vertex.label = "",  
 edge.arrow.size = 0.5,  
 vertex.size = 20)  
  
# Figure 2  
  
  
  
par(mar=c(0,0,0,0)+1)  
plot(  
 igraph\_simplified,  
 edge.label.color = "#801818",  
 edge.label = E(igraph)$label,  
 edge.label.cex = 1,  
 edge.color = "grey",  
 edge.arrow.size = 0.3,  
 vertex.size = 5,  
 vertex.shape = "square",  
 vertex.color = "orange",  
 vertex.label = V(igraph)$name,  
 vertex.label.cex = 1.0,  
 vertex.label.dist = 1.5,  
 vertex.label.degree = pi / 2,  
 edge.curved = TRUE,  
 layout = l  
)  
  
# Figure 3  
  
  
set.seed(5208)  
par(mfrow= c(1,1),mar=c(0,0,0,0)+.1)  
# Create two clusters  
cluster1 <- sample(1:10, 5, replace = FALSE)  
cluster2 <- sample(11:20, 5, replace = FALSE)  
# Create edges within clusters  
edges\_within\_cluster1 <- t(combn(cluster1, 2))  
edges\_within\_cluster2 <- t(combn(cluster2, 2))  
# Create edge connecting the clusters  
edge\_between\_clusters <- matrix(c(sample(cluster1, 1), sample(cluster2, 1)), ncol = 2)  
# Combine edges  
edges <- rbind(edges\_within\_cluster1, edges\_within\_cluster2, edge\_between\_clusters)  
# Create graph  
g <- igraph::graph\_from\_edgelist(edges, directed = FALSE)  
# Calculate betweenness centrality  
betweenness\_values <- igraph::edge\_betweenness(g)  
# Get the edge with the highest betweenness  
max\_betweenness\_edge <- which.max(betweenness\_values)  
# Set edge color  
igraph::E(g)$color <- "black"  
igraph::E(g)[max\_betweenness\_edge]$color <- "red"  
g <- igraph::induced\_subgraph(g, which(igraph::degree(g) > 0))  
# Plot the graph  
plot(  
 g,  
 vertex.label = "",  
 vertex.color = "grey",  
 edge.curved = FALSE,  
 edge.label = NA  
)  
  
# Figure 5  
  
set.seed(5208)  
par(mfrow= c(1,1),mar=c(0,0,0,0)+.1)  
  
num\_nodes <- 6  
  
# Create an empty graph  
g <- igraph::make\_empty\_graph(n = num\_nodes)  
  
# Add edges to connect all nodes to the central node (node 1)  
for (i in 2:num\_nodes) {  
 g <- igraph::add\_edges(g, c(1, i))   
}  
  
g |>  
 igraph::as.undirected()|>  
 plot(  
 vertex.label="",  
 vertex.color = ifelse(igraph::V(g)== 1, "red", "grey"),  
 edge.color = "black"  
 )  
  
  
# Figures 7-10  
  
  
# Putting this chunk here  
gn\_igraph <- igraph::cluster\_edge\_betweenness(igraph)  
  
louvain\_igraph <- igraph |>  
 igraph::as.undirected() |>   
 igraph::cluster\_louvain()  
  
sp\_igraph <- igraph |>  
 ig.degree.betweenness::cluster\_degree\_betweenness()  
  
  
# Figure 7  
  
par(mar=c(0,0,0,0)+1)  
plot(  
 gn\_igraph,  
 igraph\_simplified,  
 edge.label.color = "#801818",  
 edge.label = E(igraph)$label,  
 edge.label.cex = 1,  
 edge.color = "grey",  
 edge.arrow.size = 0.3,  
 vertex.size = 5,  
 vertex.shape = "square",  
 vertex.color = "orange",  
 vertex.label = V(igraph)$name,  
 vertex.label.cex = 1.0,  
 vertex.label.dist = 1.5,  
 vertex.label.degree = pi / 2,  
 edge.curved = TRUE,  
 layout = l  
)  
  
# Figure 8  
  
par(mar=c(0,0,0,0)+1)  
plot(  
 louvain\_igraph,  
 igraph\_simplified,  
 edge.label.color = "#801818",  
 edge.label = E(igraph)$label,  
 edge.label.cex = 1,  
 edge.color = "grey",  
 edge.arrow.size = 0.3,  
 vertex.size = 5,  
 vertex.shape = "square",  
 vertex.color = "orange",  
 vertex.label = V(igraph)$name,  
 vertex.label.cex = 1.0,  
 vertex.label.dist = 1.5,  
 vertex.label.degree = pi / 2,  
 edge.curved = TRUE,  
 layout = l  
)  
  
# Figure 9  
  
par(mar=c(0,0,0,0)+1)  
plot(  
 sp\_igraph,  
 igraph\_simplified,  
 edge.label.color = "#801818",  
 edge.label = E(igraph)$label,  
 edge.label.cex = 1,  
 edge.color = "grey",  
 edge.arrow.size = 0.3,  
 vertex.size = 5,  
 vertex.shape = "square",  
 vertex.color = "orange",  
 vertex.label = V(igraph)$name,  
 vertex.label.cex = 1.0,  
 vertex.label.dist = 1.5,  
 vertex.label.degree = pi / 2,  
 edge.curved = TRUE,  
 layout = l  
)  
  
# Figure 10  
  
all\_degree<- igraph::degree(igraph) |>   
 as.data.frame()|>  
 tibble::rownames\_to\_column()|>  
 dplyr::rename(degree=`igraph::degree(igraph)` ,  
 study=rowname)  
  
in\_degree <- igraph::degree(igraph, mode = "in")|>  
 as.data.frame()|>  
 tibble::rownames\_to\_column()|>  
 dplyr::rename(in\_degree=`igraph::degree(igraph, mode = "in")` ,  
 study=rowname)  
  
out\_degree <- igraph::degree(igraph, mode = "out") |>  
 as.data.frame()|>  
 tibble::rownames\_to\_column()|>  
 dplyr::rename(out\_degree=`igraph::degree(igraph, mode = "out")` ,  
 study=rowname)  
  
degree\_df <- merge(in\_degree,  
 out\_degree)|>  
 merge(all\_degree)|>  
 dplyr::mutate(in\_degree = -in\_degree)|>  
 tidyr::pivot\_longer(cols = c(in\_degree,out\_degree))  
  
ggplot(degree\_df,  
 mapping = aes(y =reorder(study, degree), x = -value, fill = name))+  
 theme\_minimal()+  
 geom\_col()+  
 geom\_hline(yintercept = 2.5,linetype='dashed',lwd=1)+  
 geom\_hline(yintercept = 12.5,linetype='dashed',lwd=1)+  
 theme(axis.title.y = element\_blank(),  
 legend.title = element\_blank(),  
 legend.position = "bottom",  
 axis.title.x = element\_blank())+  
 scale\_fill\_manual(labels = c("Referrals In", "Referrals Out"), values = scales::hue\_pal()(2))+  
 scale\_x\_continuous(labels = abs)  
  
# Tables  
  
# Table 1  
  
gn\_df <- data.frame(  
 Intervention = igraph::V(igraph)$name,  
 "Patient Refferalls: In" = igraph::degree(igraph,mode="in"),  
 "Patient Referrals: Out" = igraph::degree(igraph, mode="out"),  
 "Total Patient Refferals" = igraph::degree(igraph, mode="total"),  
 row.names = NULL,  
 check.names = FALSE  
) |>  
 dplyr::group\_by(Intervention) |>   
 dplyr::summarise(  
 `Refferalls In` = sum(`Patient Refferalls: In`),  
 `Referrals Out` = sum(`Patient Referrals: Out`),  
 `Total` = sum(`Total Patient Refferals`)  
 )  
  
gt::gt(gn\_df)|>  
 gt::tab\_header("Table 1: Girvan-Newman communities identified. Each intervention is their own community.")|>  
 gt::cols\_width(  
 Intervention ~ gt::pct(40),  
 `Refferalls In` ~ gt::pct(15),  
 `Referrals Out` ~ gt::pct(20),  
 `Total` ~ gt::pct(15)  
 ) |>  
 gt::tab\_options(table.font.size=42)  
  
  
  
# Table 2  
  
louvain\_df <- data.frame(  
 Intervention = igraph::V(igraph)$name,  
 Community = paste0("Community: ", igraph::membership(louvain\_igraph)|> as.vector()),  
 "Patient Refferalls: In" = igraph::degree(igraph,mode="in"),  
 "Patient Referrals: Out" = igraph::degree(igraph, mode="out"),  
 "Total Patient Refferals" = igraph::degree(igraph, mode="total"),  
 row.names = NULL,  
 check.names = FALSE  
)  
  
louvain\_df |>  
 dplyr::group\_by(Community,Intervention) |>   
 dplyr::summarise(  
 `Refferalls In` = sum(`Patient Refferalls: In`),  
 `Referrals Out` = sum(`Patient Referrals: Out`),  
 `Total` = sum(`Total Patient Refferals`)  
 )|>  
 gt::gt()|>  
 gt::tab\_header("Table 2: Louvain communities identified and grouped interventions.")|>  
 gt::cols\_width(  
 Intervention ~ gt::pct(40),  
 `Refferalls In` ~ gt::pct(15),  
 `Referrals Out` ~ gt::pct(20),  
 `Total` ~ gt::pct(15)  
 )|>  
 gt::tab\_options(table.font.size=42)  
  
  
  
# Table 3  
  
sp\_df <- data.frame(  
 Intervention = igraph::V(igraph)$name,  
 Community = paste0("Community: ", igraph::membership(sp\_igraph)|> as.vector()),  
 "Patient Refferalls: In" = igraph::degree(igraph,mode="in"),  
 "Patient Referrals: Out" = igraph::degree(igraph, mode="out"),  
 "Total Patient Refferals" = igraph::degree(igraph, mode="total"),  
 row.names = NULL,  
 check.names = FALSE  
)  
  
sp\_df |>  
 dplyr::group\_by(Community,Intervention) |>   
 dplyr::summarise(  
 `Refferalls In` = sum(`Patient Refferalls: In`),  
 `Referrals Out` = sum(`Patient Referrals: Out`),  
 `Total` = sum(`Total Patient Refferals`)  
 )|>  
 gt::gt()|>  
 gt::tab\_header("Table 3: Smith-Pittman communities and identified and grouped interventions.")|>  
 gt::cols\_width(  
 Intervention ~ gt::pct(40),  
 `Refferalls In` ~ gt::pct(15),  
 `Referrals Out` ~ gt::pct(20),  
 `Total` ~ gt::pct(15)  
 )

1. For more information, see https://www.canada.ca/en/health-canada/services/clinical-trials.html [↑](#footnote-ref-1)
2. Quality assurance studies in the context of medical studies are studies which look at drugs which are already approved for use but the goals are focused on other aspects of care such as drug delivery or quality of care. [↑](#footnote-ref-2)
3. Named after the author and his co-supervisor, Tyler Pittman. [↑](#footnote-ref-3)
4. Work on extending the Louvain algorithm to accommodate directed graphs has been an outstanding issue in the igraph community since 2015 (See: <https://github.com/igraph/igraph/issues/890>). However, Dugué and Perez12 (2022) have done some work on this. [↑](#footnote-ref-4)
5. Short for Monoclonal Antibodies. [↑](#footnote-ref-5)