The Effects of Network Rescaling

Mitchell Joseph[1] and James Bagrow[1,2,*]

[1]Vermont Complex Systems Center, University of Vermont, Burlington, VT, United States
[2]Mathematics & Statistics, University of Vermont, Burlington, VT, United States
[*]Corresponding author. Email: james.bagrow@uvm.edu, Homepage: bagrow.com

April 10, 2025

Abstract Networks have long been used to study socio-techno systems [1]. One technique that has enhanced the study of these networks has been the application of spectral analysis and matrix factorization [2, 3, 4] to obtain a rescaled version of the network. Once this rescaled version of the network has been obtained it is possible to use dimensionality reduction techniques to visualize the results [5, 6, 7, 8], which have been shown to have some success in identifying certain social behaviors. An extension of the work done by Bagrow and Lehmann [9] began to explore the effects of these rescaling methods, however an open question still remained as to why these methods seemingly worked well for certain networks but not others. The purpose of this study is a continuation of that work - to explore the question of when is it beneficial to rescale a network.

1 Introduction

The study of social networks can be dated back to the 1930s with Jacob Moreno's work *Who Shall Survive?*, which explored the social dynamics between groups of individuals [10]. This seminal paper has been attributed with being the foundation from which social network analysis was born [1]. While network science has a rich history of being used in sociological studies, it didn't take long for before researchers found their application could extend to biological and technological systems as well [11].

As researchers began to realize the advantages of using network science as a tool to enhance their studies, the field began to blossom. Originally, many studies focused on static networks, a single snapshot of the network in time, mainly for their simplicity. However, by shifting the focus to temporal networks, researchers were able to capture more complex dynamics and emergent behavior [12, 13, 14].

Other areas of research that have grown recently have been the exploration of community detection and clustering algorithms [15, 16, 17]. Earlier methods focused on using spectral clustering [2] to obtain partitions of various types of networks [3, 4]. We'll explore spectral clustering in more detail in section 2.2, but in short it leverages the eigenvectors of the Laplacian to find natural partitions within the original graph. More recently the use of stochastic blockmodels [18, 19] have been shown to be effective at a multitude of clustering tasks like community detection [20], core-periphery identification [21], and imperfect graph coloring [22].

Traditionally though, most of the development around community detection has focused on clustering from the node level of a network. However, this has been shown to have drawbacks as nodes can belong to multiple groups, thereby making it difficult to ascertain the overall hierarchical structure that typically exists. It has been shown that by shifting the focus from communities of nodes to communities of links it is possible to identify these hierarchical structures even when there is overlap between communities [23, 9].

2 Background

2.1 Temporal Networks

Most networks can be represented as a graph $G = (V, E)$ where V is the set of vertices and E the set of edges. However, this representation only gives a static snapshot of the network. While this may be an appropriate representation in some cases, most networks are dynamic - constantly changing and evolving over time. There are many examples in which temporal networks are advantageous to use such as person-to-person communication, one-to many information dissemination, physical proximity, cell biology, distributed computing, infrastructure networks, neural and brain networks, ecological networks amongst many others [14, 12]. There are two methods often used for representing temporal networks: (i) contact sequences and (ii) interval graphs. Contact sequences start by assuming that time T is negligible and define a set of contacts $C = (i, j, t)$ where $i, j \in V$ and t denotes the time. Since we're defining an interaction between two nodes, we can equivalently define this as a set of M links, where each link represents a pair of vertices. Interval graphs assume that time is non-negligible and capture the activity of edges over a set of time intervals instead of a set of individual times. By using temporal networks we are able to capture the underlying dynamical processes inherent in most networks [13].

2.2 Spectral Clustering

Spectral clustering leverages linear algebra, specifically the eigenvectors of the Laplacian, to find natural structure within graphs [24, 25, 26, 2]. Real world use cases have applied spectral clustering to segment images [3] and extensions have been made to apply these methods to bipartite graphs as well [4].

Traditionally, these methods start by letting $G = (V, E)$ be an undirected weighted graph with vertices $v \in V$ and an edge between nodes v_i and v_j is defined as having a strictly non-negative weight $w_{ij} \geq 0$. We can then define the adjacency matrix of the graph $\boldsymbol{W}$ whose entry ij is equal to w_{ij}. From here we can define the degree matrix $\boldsymbol{D}$, a diagonal matrix whose entries are the degree of each vertex. The degree of v_i is defined as $d_i = \sum_{j=1}^{n} w_{ij}$ [27].

Now that we have both our adjacency matrix and degree matrix we can define both the unnormalized and normalized graph Laplacian matrices. The unnormalized Laplacian, often times referred to simply as the Laplacian of a graph, is defined as $L = D - W$. The normalized varieties are the symmetric Laplacian $L_{sym} := D^{-1/2}WD^{-1/2}$ and the random walk Laplacian $L_{rw} := D^{-1/2}W$. When we talk about rescaling a network, these are the transformations that we will be referring to. However, instead of an adjacency matrix we will make use of the incidence matrix, which is an $M \times T$ matrix where M is the number of links and T is the corresponding time period. The entries of this matrix are the number of interactions observed for a given link during the time interval. This will be discussed in further detail in section 3.2. Once a rescaled version of the network has been obtained, it has been noted that using the eigenvector with the second smallest eigenvalue, also known as the Fiedler value, can be used to partition the graph [28, 4].

3 Materials and Methods

3.1 Dataset

This study focused on five separate networks: Copenhagen, Hospital, Ant Colony, Manufacturing Email, and College Message.

1. **Copenhagen** A temporal network capturing the interactions between 700 first year students at the University of Denmark [29, 30]. An interaction was defined by using bluetooth to assess if two participant's smartphones were within close proximity to one another. Measurements were taken every 5 minutes across 36 months. For our use case, the time was aggregated into 1h bins and links with fewer than 200 interactions were excluded. The final result was a dataset with 544 nodes, 2389 edges, and 672 time periods.

2. **Hospital** A temporal network looking at interactions between patients and hospital staff [31]. A contact was said to be made if two people were within 1 meters of each other and measurements were taken every 20 seconds. The original authors noted a natural periodicity with 94.1% of the encounters occurring during the day and 5.9% at night. The data collection period spanned from 1pm Monday December 6, 2010 to 2pm Friday December 10, 2010. Processing steps included binning the data into 1h increments and edges were required to have a minimum of 20 interactions after which the network consisted of 68 nodes, 329 edges, and 96 time periods.

3. **Ant Colony** A temporal network looking at the dynamics of individual interactions amongst ants [32]. An ant was said to have interacted with another ant if the antenna of one made contact with the body of the other. The ants were also marked with different colors to uniquely identify which colony they belonged to. Interactions between ants were recorded during 30 minute intervals amongst a total of four different colonies. The data we used had been processed to have a minimum of three interactions and were aggregated into 10 second bins. We were left with a network of 85 nodes, 269 edges, and 143 time periods.

4. **Manufacturing Email** A temporal network focusing on social network hierarchy and dynamics through the analysis of corporate emails [33]. The manufacturing company comprised of 300 employees, 1/3 of which were clerical workers and the rest were laborers. However, only a subset of the organization was used for the study. In total there were 11,816 emails spanning from January 1, 2010 to September 30, 2010. During preprocessing it was required that a minimum of 10 interactions be observed during the period and the time was aggregated into 1 day bins. In the end we were left with 140 nodes, 1137 edges, and 273 time periods

5. **College Message** A temporal network analyzing the online messages sent between college students at the University of California Irvine [34]. The network was observed from April to October of 2004 and includes all users who sent or received at least one message. A total of 1,899 users were registered and the start of each sample began at 7:00am to correspond to the lowest usage. The preprocessing steps taken for our use case involved ensuring a minimum of 10 interactions and aggregating the observable periods into 1 day bins. We were then left with 609 nodes, 1,323 edges and 194 time periods.

We use the methodology implemented by Tang et. al. to represent each network as a link activity matrix $\boldsymbol{E} \in \mathbb{R}^{M \times T}$ where each row represents a link and each column is a time period [35]. Therefore we have $E_{ij,t}$ is the number of interactions for link ij at time t. The number of links and time periods can be seen in Table 1 below.

Network	Links (M)	Time Periods (T)
Copenhagen	2389	672
Hospital	329	96
Ant Colony	269	143
Manufacturing Email	1137	273
College Message	1323	194

Table 1: Number of links and time periods observed for each network after processing - including removing observations with too few interactions and rescaling the time bins.

3.2 Methods

In our attempts to better understanding how rescaling affects the overall structure of the networks we analyzed several permutations of the networks. As noted above all of the networks studied here have been represented as link activity matrices, which we will refer to simply as the link matrix denoted as E. It follows that by taking the transpose of E we are left with a time matrix T. Other permutations on these two matrices included duplicating, or stacking, each matrix. So for example if $E \in \mathbb{R}^{M\times T}$ then we could duplicate along either axis to yield a time duplicate of the links which would have shape $M \times 2T$, or alternatively we could do a link duplication of the link matrix which would be of shape $2M \times T$. Similarly we could do the same and have a link dupe of the time matrix which would be $T \times 2M$ or a time dupe of the time matrix which would be of shape $2T \times M$.

In this paper we attempted to use both the symmetric and random walk Laplacians as described in 2.2. However, since the data being used in this research is sparse special care needed to be taken to avoid singular matrices particularly when analyzing T. For example, it happens to be the case that $t_j \mathbf{0}$ for some time periods j, which corresponds to there being a time in the study in which no interactions were observed. However, this leads to a singular matrix, so to handle these instances we propose the following steps.

Let $\boldsymbol{E} \in \mathbf{R}^{M\times T}$ be a matrix with M links and T time periods. Then we let $R \in \mathbf{0}_{M,T}$ be the matrix of zeros with shape $M \times T$. From here we take the column sums $c = \boldsymbol{E}$, now c acts as an index and we can create a new matrix $\hat{\boldsymbol{E}}$ that contains rows only where the column sums c is not equal to zero. We observe that $\hat{\boldsymbol{E}}$ is now non-singular and so we can create the respective diagonal matrices $\boldsymbol{D_X}$ and $\boldsymbol{D_Y}$ from the row and column sums respectively. Now we apply the symmetric normalization as outlined before $\boldsymbol{E_{res}} = D_X \hat{\boldsymbol{E}} D_Y$, and from here we insert this matrix back into our zero matrix $\boldsymbol{R}$ everywhere where that $c > 0$. This will yield the rescaled matrix using the symmetric normalization, and only a slight modification is needed to obtain the random walk variant.

Once a rescaled version of the matrix had been obtained we then used several dimensionality reduction techniques [36] including: T-SNE [7, 37], UMAP [6], TriMAP [8], and PaCMAP [5] to visualize the both the original and rescaled versions of the network.

4 Results

4.1 Preliminary Results

We began by recreating some of the original work done by Bagrow and Lehmann [9]. They had showed that the Copenhagen network seemed to benefit the most from rescaling, particularly when focusing on the time embedding of the network. Below we see three plots, all of which contain the time embedding of the original Copenhagen network on the left and the rescaled version on the right. We begin with Fig. 1 and note that by using the left singular vector as outlined by Zha [4] to partition the data we are able to get a clear distinction between the central mass and the outer tendrils. By then shifting to highlighting the points based off of weekdays and weekends (Fig. 2) we can see that the tendrils of the rescaled network correspond nearly perfectly with the weekends. However, at this time we noted that the original network does not benefit in the same way, and while some of the points that are further away from the center of mass do correspond to weekends, it isn't as clearly defined as the rescaled version. We partitioned the data one last time to get even more granularity by breaking down the weekends into their respective days and evenings (Fig. 3). This highlighted an interesting observation that the Sunday evenings were closer to weekdays than other times suggesting that most students in the study treated a Sunday evening more like a typical school day. However, we again note that this is visible most clearly in the rescaled version, and less so in the original network.

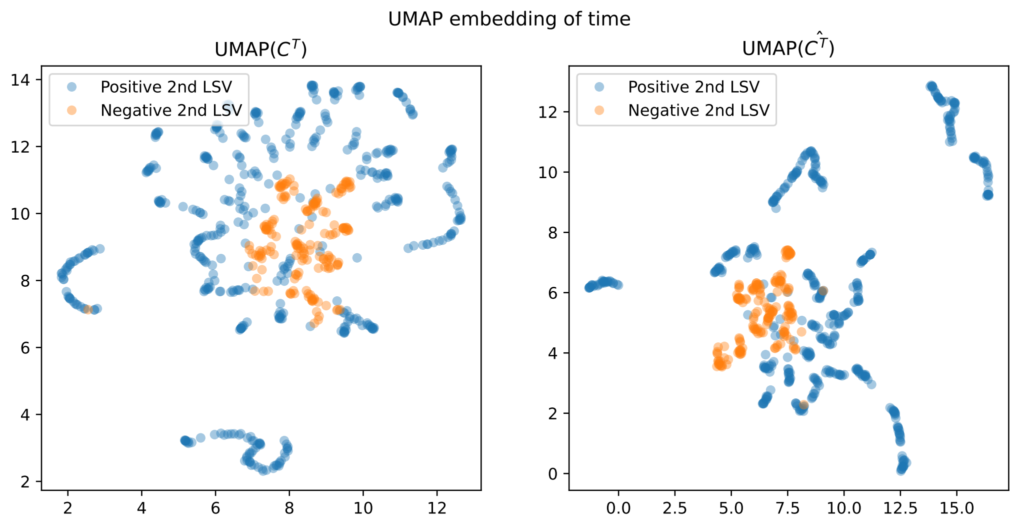

Figure 1: UMAP embedding of time for the original Copenhagen network (C) and the rescaled network $\hat{C}$ - partitioned using the left singular vector.

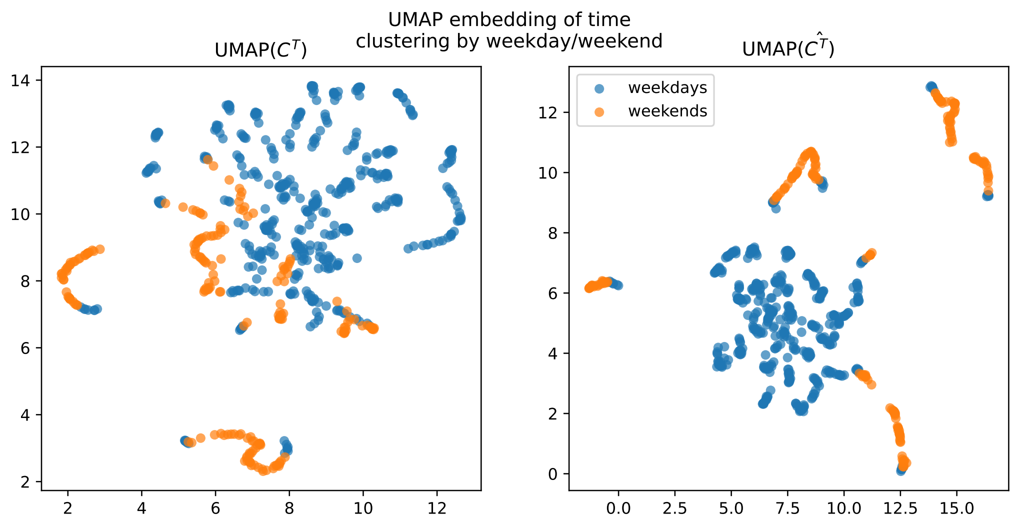

Figure 2: UMAP embedding of time for the original Copenhagen network (C) and the rescaled network $\hat{C}$ with weekends highlighted in orange and weekdays highlighted in blue.

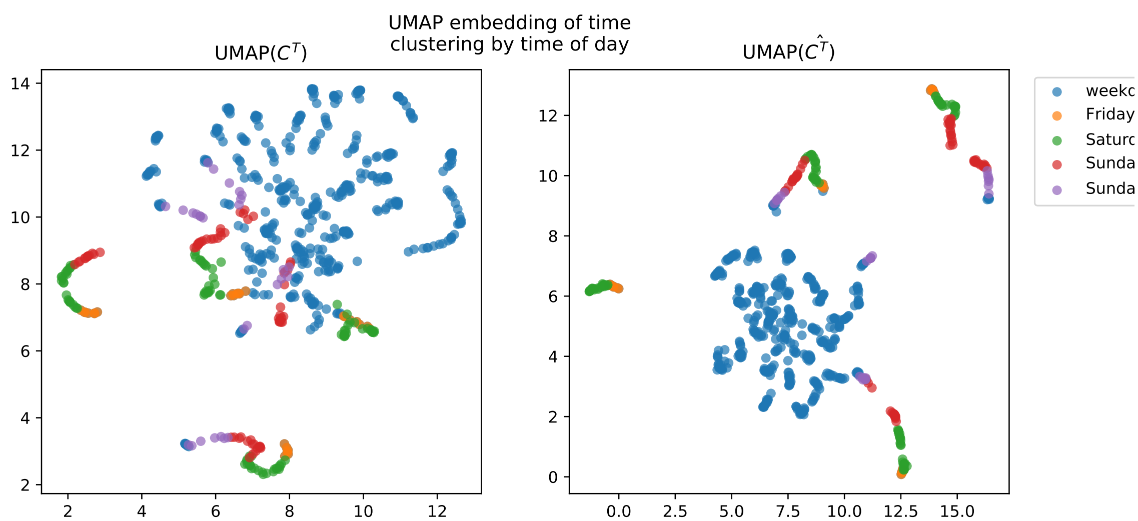

Figure 3: UMAP embedding of time for the original Copenhagen network (C) and the rescaled network $\hat{C}$ with days of the week highlighted.

4.2 Duplicating the Network

As seen above, when rescaling the time matrix, we can clearly identify the weekends within the tendrils of the plot. We decided to explore this further by looking at the time duplication of the time matrix for the Copenhagen network with a duplication of 0, 2, and 10, where 0 corresponds to the original network with no duplications as seen in Fig 4 and Fig 5. We chose to focus on the Copenhagen network as it was observed to benefit the most from rescaling, with our assumption being that it would be easier to detect the impact of rescaling on a network already benefiting from it as opposed to other networks which seemed to have less of a result. We also note that while UMAP seems to provide a useful separation of the weekends for the rescaled version compared to the original, this distinction breaks down as we move to more duplications. Further by the time we get to ten duplications the distinctions become completely indecipherable. We had hoped that by duplicating the network we would be able to make the distinctions between weekends and weekdays more pronounced, however, the opposite effect appears to have occurred. We also note that TriMAP was unable to provide any meaningful insight, even without any duplications within the network, and while PaCMAP showed some promise of being useful for the original network it was less beneficial in the rescaled instance, which opened more questions as to why rescaling would be beneficial.

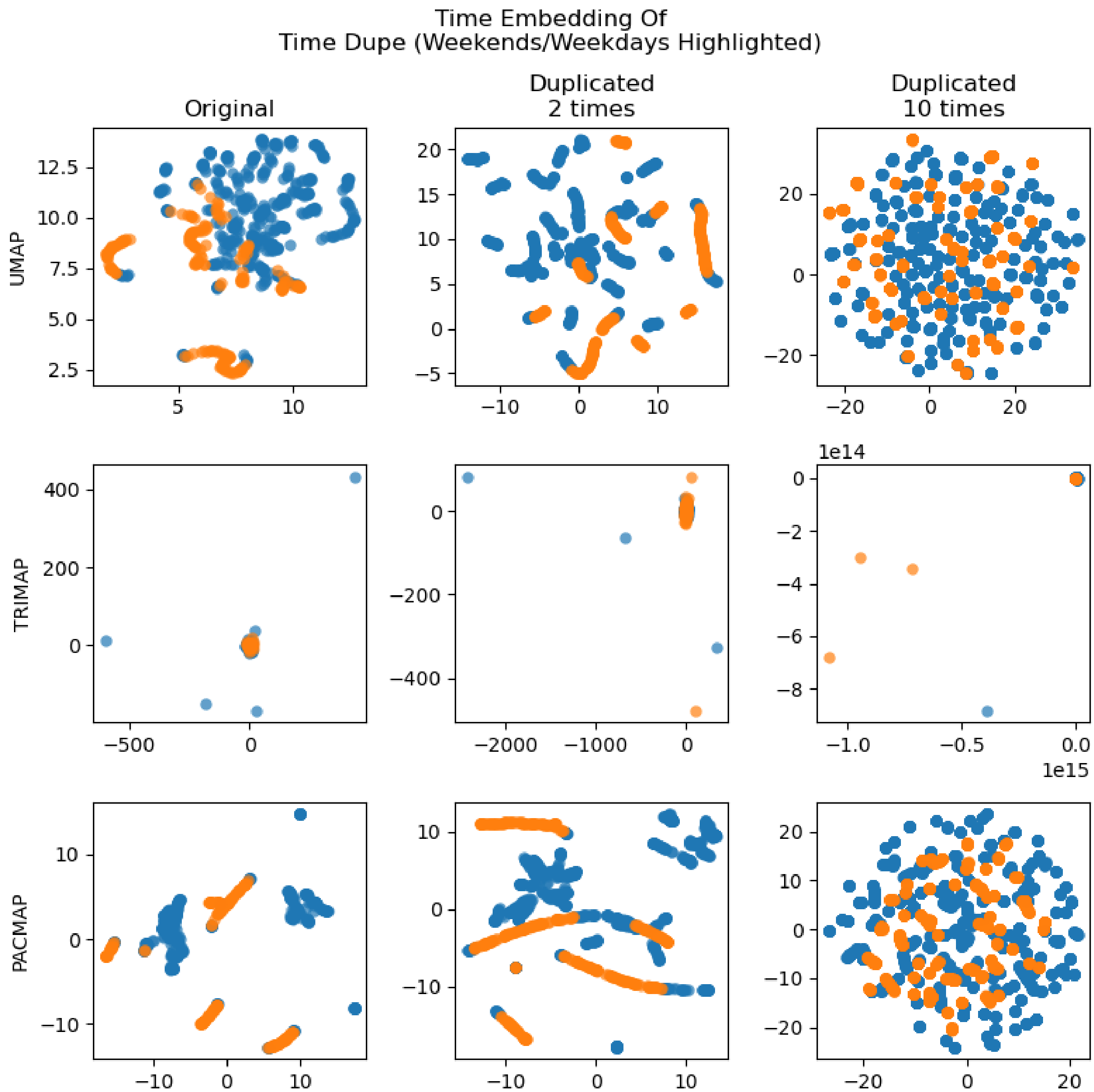

Figure 4: Time embedding of the time duplicated Copenhagen network with no rescaling. Weekends are highlighted in orange with weekdays highlighted in blue.

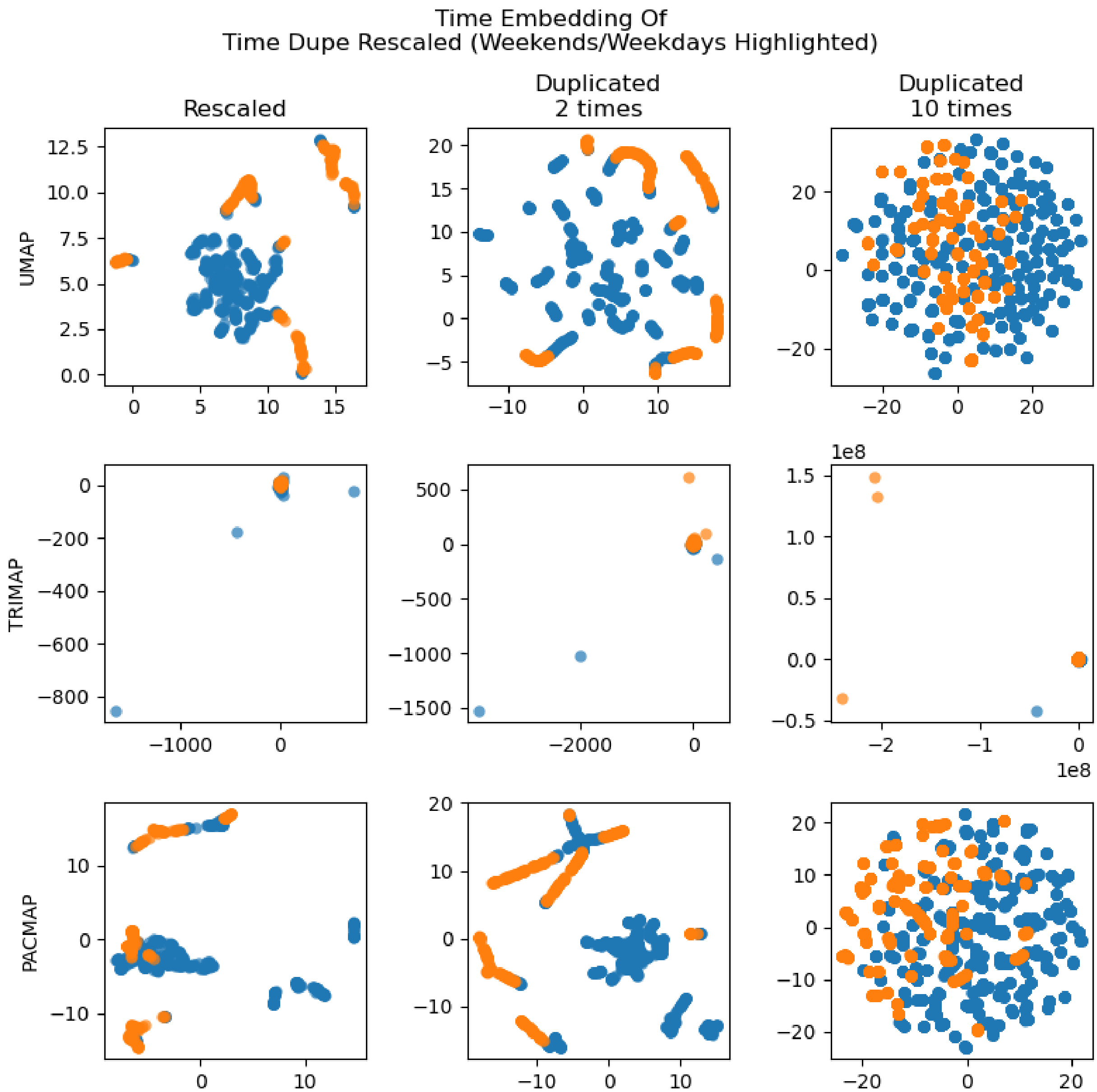

Figure 5: Time embedding of the time duplicated Copenhagen network rescaled. Weekends are highlighted in orange with weekdays highlighted in blue

4.3 Adding Noise

After duplicating the data, we decided to see what would happen if we added a small amount of noise by multiplying each value in our matrix by a random number drawn from $\sim N(1, 0.1)$. By multiplying instead of adding noise to our data, we preserve the sparsity while slightly modifying the observed values. We see below in Fig. 6 and Fig. 7 that there is still no clear benefit from duplicating, even when adding noise. TriMAP did perform better this time, even when duplicated twice which was interesting to note compared to its lackluster performance before. However, like all the embedding methods, once we move out to ten duplications of the network it becomes too challenging to distinguish weekdays from weekends.

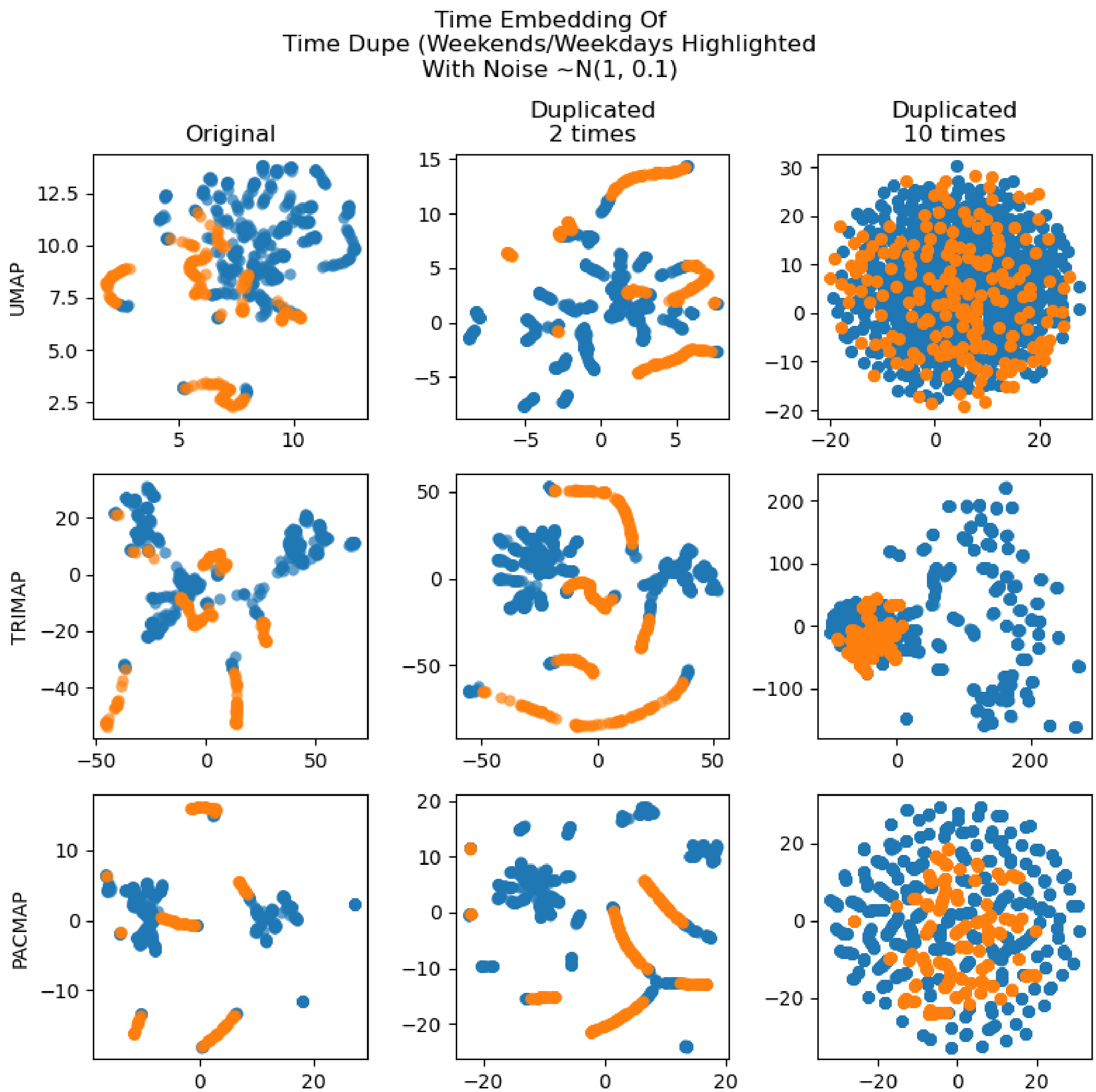

Figure 6: The time embedding of the original Copenhagen network along with the duplication of time 2 and 10 times. Weekends are highlighted in orange with weekdays highlighted in blue.

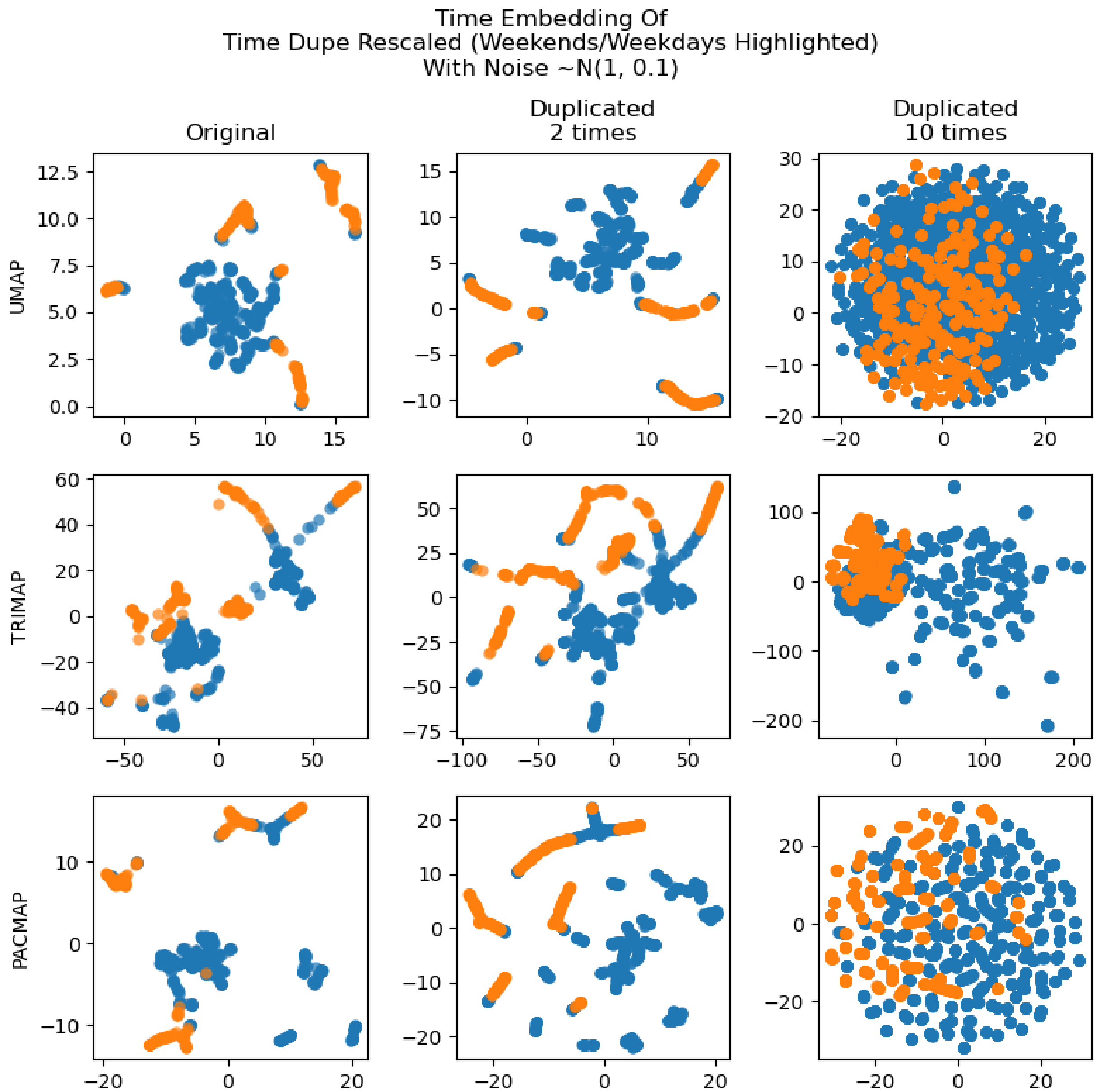

Figure 7: The time embedding of the rescaled Copenhagen network along with the duplication of time 2 and 10 times. Weekends are highlighted in orange with weekdays highlighted in blue.

4.4 Artificial Data

With our previous attempts failing to unearth any meaningful insight, we shifted our attention to creating a synthetic network to try and induce a distinct cyclical behavior, with the idea being we should be able to better identify how the rescaling methods affect these networks. Since all of our networks were based on encounters, we decided to use two

Poisson distributions to populate an artificial network meant to replicate an exaggerated version of the Copenhagen study. We set $\lambda_1 \sim Poisson(3)$ and $\lambda_2 \sim Poisson(15)$, which we felt was a large enough difference between the distributions to create a sizable difference when constructing the data. Similar to the Copenhagen data we set our time intervals to be every hour and so to create a weeks worth of simulated data we populated an $n \times 48$ sparse matrix with values drawn from λ_1 and an $n \times 120$ sparse matrix with values drawn from λ_2. Using this function, we could then create months and years of artificial data. Since UMAP seemed to be performing the best out of the embedding methods for our use case, we decided to test how UMAP handled data when the number of observations was much less than the number of time periods ($n << p$). For this we generated a dataset with 100 observations and $8,064$ time periods. The results of this embedding can be seen in Fig. 8 where we again highlight the weekends in orange and weekdays in blue. We then replicate a similar design but flipped the dimensionality so $n >> p$. For this we simulated $10,000$ observations across 672 time periods, the results of which can be seen in Fig. 9. We were surprised to find that even when simulating data with a large disparity between groups that neither embedding method was able to accurately distinguish between the two groups.

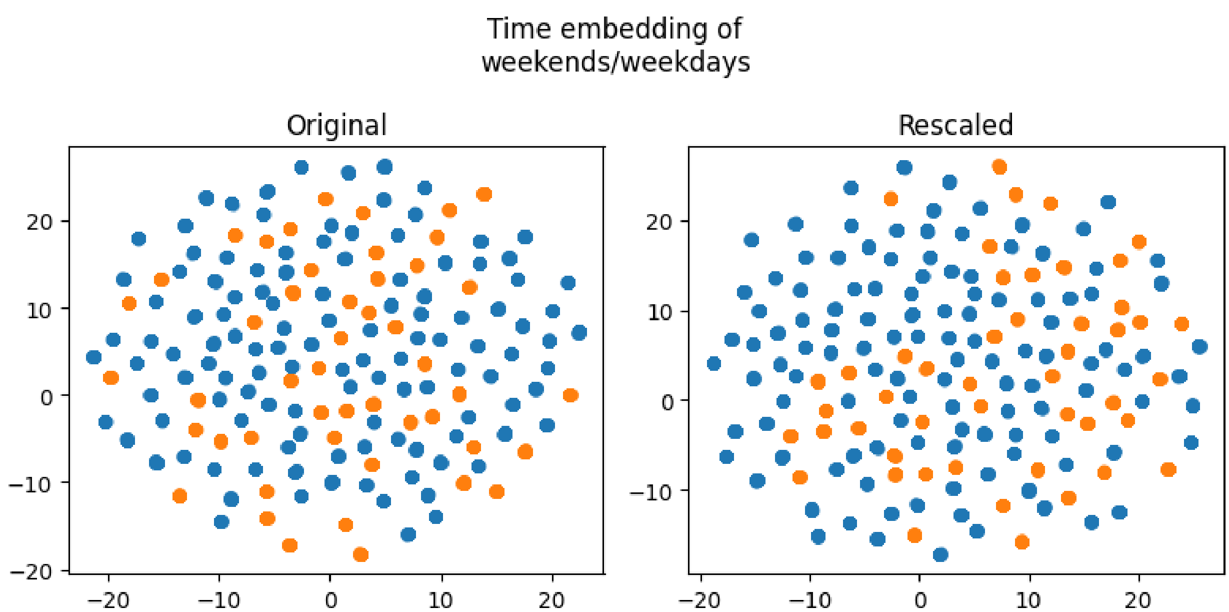

Figure 8: UMAP embedding of an artificial dataset with n=100 and t=8,064. Weekdays are highlighted in blue with weekends highlighted in orange.

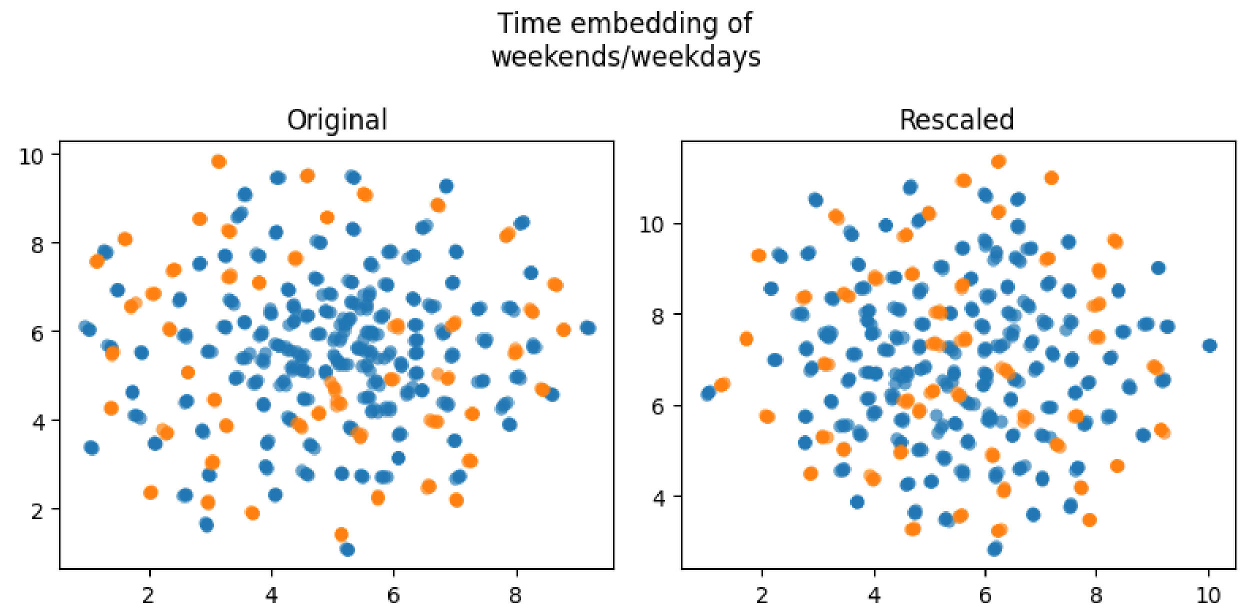

Figure 9: UMAP embedding of an artificial dataset with n=10,000 and t=672. Weekdays are highlighted in blue with weekends highlighted in orange.

5 Discussion

Throughout this paper we tried to explore the question of when is it beneficial to rescale a network. We explored some preliminary results along with what happens when we try these embedding methods on duplications of the original network in an attempt to extract underlying patterns. When that didn't work we then attempted to create an artificial dataset to manually induce the desired behavior, however those attempts also fell short. We chose to focus the results of this report on the Copenhagen network as that had the most promising results, but we continued our analysis on the remaining networks described in section 3.1, which can be found in the supplementary material[1]. The supplementary material also covers the random walk rescaling, rescaling the links, we also performed these analyses on the time embedding on links and link embeddings on time, we tried truncating the time periods to see if the embeddings were influenced by the number of cycles, we tried visualizing the plots of the embedded network in three dimensions to see if we were losing information by reducing the dimensionality too much, we also permuted the rows and columns to see how that effected the embedding, and lastly tried embedding the network augmented with the incidence matrix. All of these attempts failed to yield any meaningful insights. While it is always disappointing to not come away with any results, we acknowledge that this is part of the scientific process, and sometimes null results are still results worth sharing. We hope that this work can serve as inspiration to others to continue to explore these embedding methods,

[1]https://github.com/baglab/times-ties-activity-data

and determine when is it appropriate to utilize them.

6 Future Work

There are still many questions that remain unsolved. Are there any metrics that would help determine when rescaling is beneficial? Possible areas of exploration are the distribution of singular values, comparing the distribution of singular vectors, the relation between the row sums and column sums, the mean edge activity over time, the sparsity (the number of active links at time t). These are just a few of the questions that we suggest still warrant exploration. It has also been shown that there is a link between correspondence analysis and spectral clustering and graph-embedding techniques [38]. While we didn't have enough time to incorporate correspondence analysis into our work here, it is possible that there could be answers within that field.

Acknowledgments

Thank you to Dr. Bagrow for his advisement and help throughout this project.

References

[1] M. Newman, *Networks*. Oxford university press, 2018. 1

[2] F. R. Chung, *Spectral graph theory*, vol. 92. American Mathematical Soc., 1997. 1, 2

[3] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 888–905, 2000. 1, 2

[4] H. Zha, X. He, C. Ding, H. Simon, and M. Gu, "Bipartite graph partitioning and data clustering," in *Proceedings of the tenth international conference on Information and knowledge management*, pp. 25–32, 2001. 1, 2, 3, 6

[5] Y. Wang, H. Huang, C. Rudin, and Y. Shaposhnik, "Understanding how dimension reduction tools work: an empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization," *The Journal of Machine Learning Research*, vol. 22, no. 1, pp. 9129–9201, 2021. 1, 5

[6] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018. 1, 5

[7] L. Van der Maaten and G. Hinton, “Visualizing data using t-sne.,” *Journal of machine learning research*, vol. 9, no. 11, 2008. 1, 5

[8] E. Amid and M. K. Warmuth, “Trimap: Large-scale dimensionality reduction using triplets,” *arXiv preprint arXiv:1910.00204*, 2019. 1, 5

[9] J. P. Bagrow and S. Lehmann, “Recovering lost and absent information in temporal networks,” *arXiv preprint arXiv:2107.10835*, 2021. 1, 2, 5

[10] J. L. Moreno, “Who shall survive?: A new approach to the problem of human interrelations.,” 1934. 1

[11] K.-C. Chen, M. Chiang, and H. V. Poor, “From technological networks to social networks,” *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 9, pp. 548–572, 2013. 1

[12] N. Masuda and R. Lambiotte, *A guide to temporal networks*. World Scientific, 2016. 1, 2

[13] A. Li, S. P. Cornelius, Y.-Y. Liu, L. Wang, and A.-L. Barabási, “The fundamental advantages of temporal networks,” *Science*, vol. 358, no. 6366, pp. 1042–1046, 2017. 1, 2

[14] P. Holme and J. Saramäki, “Temporal networks,” *Physics reports*, vol. 519, no. 3, pp. 97–125, 2012. 1, 2

[15] S. Fortunato, “Community detection in graphs,” *Physics reports*, vol. 486, no. 3-5, pp. 75–174, 2010. 1

[16] M. Girvan and M. E. Newman, “Community structure in social and biological networks,” *Proceedings of the national academy of sciences*, vol. 99, no. 12, pp. 7821–7826, 2002. 1

[17] M. E. J. Newman, “Network structure from rich but noisy data,” *Nature Physics*, vol. 14, pp. 542–545, mar 2018. 1

[18] B. Karrer and M. E. J. Newman, “Stochastic blockmodels and community structure in networks,” *Physical Review E*, vol. 83, jan 2011. 2

[19] J.-G. Young, A. Kirkley, and M. E. J. Newman, “Clustering of heterogeneous populations of networks,” *Physical Review E*, vol. 105, jan 2022. 2

[20] M. E. Newman, “Community detection in networks: Modularity optimization and maximum likelihood are equivalent,” *arXiv preprint arXiv:1606.02319*, 2016. 2

[21] S. P. Borgatti and M. G. Everett, “Models of core/periphery structures,” *Social networks*, vol. 21, no. 4, pp. 375–395, 2000. 2

[22] J.-G. Young, G. St-Onge, P. Desrosiers, and L. J. Dubé, "Universality of the stochastic block model," *Physical Review E*, vol. 98, sep 2018. 2

[23] Y.-Y. Ahn, J. P. Bagrow, and S. Lehmann, "Link communities reveal multiscale complexity in networks," *nature*, vol. 466, no. 7307, pp. 761–764, 2010. 2

[24] J. Cheeger, "A lower bound for the smallest eigenvalue of the laplacian," in *Problems in Analysis: A Symposium in Honor of Salomon Bochner (PMS-31)*, pp. 195–200, Princeton University Press, 2015. 2

[25] W. E. Donath and A. J. Hoffman, "Lower bounds for the partitioning of graphs," *IBM Journal of Research and Development*, vol. 17, no. 5, pp. 420–425, 1973. 2

[26] M. Fiedler, "A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory," *Czechoslovak mathematical journal*, vol. 25, no. 4, pp. 619–633, 1975. 2

[27] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, pp. 395–416, 2007. 3

[28] M. Fiedler, "Algebraic connectivity of graphs," *Czechoslovak mathematical journal*, vol. 23, no. 2, pp. 298–305, 1973. 3

[29] P. Sapiezynski, A. Stopczynski, D. D. Lassen, and S. Lehmann, "Interaction data from the copenhagen networks study," *Scientific Data*, vol. 6, no. 1, p. 315, 2019. 3

[30] V. Sekara, A. Stopczynski, and S. Lehmann, "Fundamental structures of dynamic social networks," *Proceedings of the national academy of sciences*, vol. 113, no. 36, pp. 9977–9982, 2016. 3

[31] P. Vanhems, A. Barrat, C. Cattuto, J.-F. Pinton, N. Khanafer, C. Régis, B.-a. Kim, B. Comte, and N. Voirin, "Estimating potential infection transmission routes in hospital wards using wearable proximity sensors," *PloS one*, vol. 8, no. 9, p. e73970, 2013. 3

[32] B. Blonder and A. Dornhaus, "Time-ordered networks reveal limitations to information flow in ant colonies," *PloS one*, vol. 6, no. 5, p. e20298, 2011. 4

[33] R. Michalski, S. Palus, and P. Kazienko, "Matching organizational structure and social network extracted from email communication," in *Business Information Systems: 14th International Conference, BIS 2011, Poznań, Poland, June 15-17, 2011. Proceedings 14*, pp. 197–206, Springer, 2011. 4

[34] P. Panzarasa, T. Opsahl, and K. M. Carley, "Patterns and dynamics of users' behavior and interaction: Network analysis of an online community," *Journal of the American Society for Information Science and Technology*, vol. 60, no. 5, pp. 911–932, 2009. 4

[35] D. Tang, W. Du, L. Shekhtman, Y. Wang, S. Havlin, X. Cao, and G. Yan, "Predictability of real temporal networks," *National science review*, vol. 7, no. 5, pp. 929–937, 2020. 4

[36] T. Hastie, R. Tibshirani, and M. Wainwright, *Statistical learning with sparsity: the lasso and generalizations*. CRC press, 2015. 5

[37] M. Wattenberg, F. Viégas, and I. Johnson, "How to use t-sne effectively," *Distill*, vol. 1, no. 10, p. e2, 2016. 5

[38] A. van Dam, M. Dekker, I. Morales-Castilla, M. Á. Rodríguez, D. Wichmann, and M. Baudena, "Correspondence analysis, spectral clustering and graph embedding: applications to ecology and economic complexity," *Scientific reports*, vol. 11, no. 1, p. 8926, 2021. 15