Multi-Graph Explorer: A Framework for Advanced Multi-Graph Analysis and Method Development

Yorgos Tsitsikas
Department of Computer Science and Engineering
University of California, Riverside
Riverside, CA, USA
gtsit001@ucr.edu

Evangelos E. Papalexakis
Department of Computer Science and Engineering
University of California, Riverside
Riverside, CA, USA
epapalex@cs.ucr.edu

Abstract

In recent years, multi-graphs have been gaining increasing popularity due to their ability to better capture the multi-faceted information of real-world graphs. This, in turn, has consistently provided superior insights and performance in related machine learning tasks. However, the analysis of real-world multi-graphs and the development of multi-graph methods is currently stifled by a few limitations. On one side, researchers often struggle to properly evaluate the performance of multi-graph methods they design due to a lack of high-quality benchmarks, but also a lack of tools that allow for efficient and seamless experimentation. On the other side, practitioners aiming to analyze real-world multi-graphs often struggle obtaining robust insights due to a lack of high-quality multi-graph methods. To this end, we present Multi-Graph Explorer: a MATLAB software designed to offer a user-friendly yet comprehensive, flexible, and extensible workflow for multi-graph analysis, aiming to break these barriers and accelerate progress in machine learning tasks involving multi-graphs.

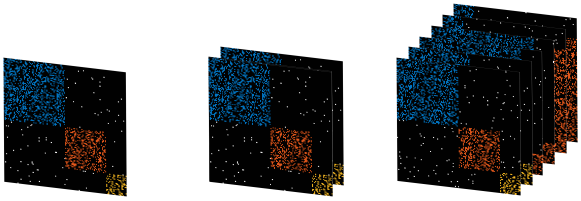
CCS Concepts

• **Mathematics of computing** → **Graph algorithms**; *Exploratory data analysis*; *Cluster analysis*; • **Human-centered computing** → *Graphical user interfaces*; **Visualization toolkits**; **User interface toolkits**; *Social network analysis*; • **Computing methodologies** → **Cluster analysis**; *Factorization methods*; *Spectral methods*; **Scientific visualization**; • **Applied computing** → *Biological networks*.

Keywords

multi-graph; multi-modal graph; multi-view graph; multi-layer graph; multi-aspect graph; multi-plex graph; tensor factorization

ACM Reference Format:
Yorgos Tsitsikas and Evangelos E. Papalexakis. 2024. Multi-Graph Explorer: A Framework for Advanced Multi-Graph Analysis and Method Development. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3627673.3679213



(a) Single-view graph **(b) Multi-view graph** **(c) Multi-structure Multi-view graph**

Figure 1: Examples of different types of graphs visualized as a series of adjacency matrices, each corresponding to a different view. Darker colors represent lower edge weights, and all edges connecting nodes in the same node cluster have the same color hue.

1 Introduction

Due to their superior modelling capabilities, graphs that allow for more than one type of edge, also known as multi-graphs, have been steadily becoming a crucial tool for properly modelling the increasing complexity of real-world graphs [6]. Multi-view graphs are a popular subtype of multi-graphs where all nodes can be related with each other by all edge types, and each sub-graph corresponding to a specific edge type is called a graph view. Important real-world examples of such graphs include time-evolving graphs where the connections between nodes are altered as time passes [10], and multi-relational knowledge graphs which aim to describe the different types of relationships between entities [8]. Also, although the relevant literature has for the most part focused on graphs whose views all have the same structure as depicted in Figure 1b [5],[4],[2],[9], there has recently been increased research activity on multi-view graphs with the more complex structures as depicted in Figure 1c [7],[3],[1].

However, despite this progress, the development of multi-graph modelling methods has been arguably stifled by a few important factors. First, in contrast to traditional graph modelling, there are currently very few multi-graph datasets whose structure is both complex enough and accompanied by sufficiently comprehensive ground-truth. In turn, this can pose an important obstacle in evaluating the performance of multi-graph methods. Second, the lack of appropriate development tools can impede the speed at which such methods can be developed while also making them potentially less effective and impactful due to inadequately evaluated algorithmic design choices. Particularly, design choices in data pre-processing, embedding generation, embedding post-processing, and methods

CIKM '24, October 21–25, 2024, Boise, ID, USA

ACM ISBN 979-8-4007-0436-9/24/10
https://doi.org/10.1145/3627673.3679213

for clustering the embeddings can often be hard to justify on a well-founded theoretical basis, and, therefore, researchers often resort to handwavy design choices that are mostly based on intuition. In light of this, we are presenting Multi-Graph Explorer, which is a MATLAB software with the following core features:

- **Artificial multi-graph generator**: A tool that enables the user to define and generate artificial multi-graphs in a succinct yet flexible and comprehensive manner.
- **Efficient large-scale Monte Carlo experiments with non-standard parameter sweeps**: Users can quickly and intuitively define value ranges for arbitrary value combinations of parameters and their subparameters, which enables the thorough and systematic study of the multi-graph analysis workflow. Also, note that calculating, storing and accessing the relevant results can be quite computationally expensive, and, therefore, considerable effort has been dedicated to performance optimizations.
- **High-quality method implementations and extensibility**: Users can readily experiment with optimized implementations of existing multi-graph methods, while also being able to augment any part of the workflow with their own method implementations.
- **Powerful exploration of experimental results**: We provide a modular graphical user interface consisting of a series of diverse and interconnected graph visualizations which may be further augmented or modified by the user. This tool also doubles as an interactive environment for efficiently adjusting the graph structure and the model parameters.

We make our software publicly available [1].

2 Proposed Framework

In this section, we discuss in detail the various components of our software.

2.1 Artificial Graph Generator

An important aspect of designing graph methods is the ability to generate artificial graphs that closely resemble real-world graphs. This allows researchers to better analyze and understand the behavior of their method, which could in turn enable them to improve its design. Our generator aims to provide a flexible tool for creating complex multi-view graphs with features such as:

- Distinct groups of views, each corresponding to a different clustering of nodes.
- Various types of node clusters such as clique, star etc.
- Directed and undirected edges.
- Edge sparsity and noise on various levels of resolution such as per view cluster, per view and per node cluster.

For instance, in Listing 1 we see how we can define and generate an artificial multi-structure multi-view graph similar to the one shown in Figure 1c. In this example, we define 3 view clusters with 2 views each, and 3, 2 and 2 node clusters, respectively. Also, each view cluster is assigned its own edge sparsity level. In the end, we get the adjacency tensor X whose frontal slices correspond to the adjacency matrices of the views of the graph.

[1] https://github.com/gtsitsik/multi-graph-explorer

```
size_all = { {60,40,20}, {100,20}, {20,100} };
sparsity_level_all = [0.94 0.93 0.92];
num_of_view_clusters = numel(size_all);
G = graph_tree_root;
for i = 1:num_of_view_clusters
    G.Children(i) = graph_tree_node;
    current_child = G.Children(i);
    current_child.slices_num = 2;
    current_child.noise_level = 0.01;
    current_child.sparsity_level = sparsity_level_all(i);
    num_of_node_clusters = numel(size_all{i});
    for j = 1:num_of_node_clusters
        current_child.Children(j).type = 'clique';
        current_child.Children(j).size = size_all{i}{j};
    end
end
X = create_graph(G);
```

Listing 1: Example of artificial multi-view graph generation.

2.2 Advanced Parameter Sweeps

Due to the nature of various parameters potentially having arbitrary subparameters, generating the various combinations of parameter values is not as straightforward as a traditional grid search. For instance, consider the scenario shown in Listing 2 which aims to combine an embedding generation method having 2 parameters, with 2 embedding clustering methods which have their own parameters too. In this case, all 6 possible combinations of parameter values are stored in variable comb, whose structure is similar to what someone would get by defining all individual combinations manually as shown in Listing 3. Additionally, the exploration of such complex parameter spaces is performed with high efficiency thanks to the tight integration with MATLAB's Parallel Computing Toolbox and our many optimizations around storing, retrieving and manipulating the large volumes of data produced by experiments.

```
prm.embedding_method.comclus.beta = [0.7 0.8]
prm.embedding_method.comclus.rho = 0.4
prm.clustering_method.kmeans.clusters_num = "3 2 2"
prm.clustering_method.large_inner_prod.thres = [0.7 0.9]
comb = generate_combinations(prm)
```

Listing 2: Compact generation of combinations.

```
comb(1).embedding_method.comclus.beta = 0.7
comb(1).embedding_method.comclus.rho = 0.4
comb(1).clustering_method.kmeans.clusters_num = "3 2 2"
comb(2).embedding_method.comclus.beta = 0.8
comb(2).embedding_method.comclus.rho = 0.4
comb(2).clustering_method.kmeans.clusters_num = "3 2 2"
comb(3).embedding_method.comclus.beta = 0.7
comb(3).embedding_method.comclus.rho = 0.4
comb(3).clustering_method.large_inner_prod.thres = 0.7
comb(4).embedding_method.comclus.beta = 0.8
comb(4).embedding_method.comclus.rho = 0.4
comb(4).clustering_method.large_inner_prod.thres = 0.7
comb(5).embedding_method.comclus.beta = 0.7
comb(5).embedding_method.comclus.rho = 0.4
comb(5).clustering_method.large_inner_prod.thres = 0.9
comb(6).embedding_method.comclus.beta = 0.8
comb(6).embedding_method.comclus.rho = 0.4
comb(6).clustering_method.large_inner_prod.thres = 0.9
```

Listing 3: Explicit generation of combinations.

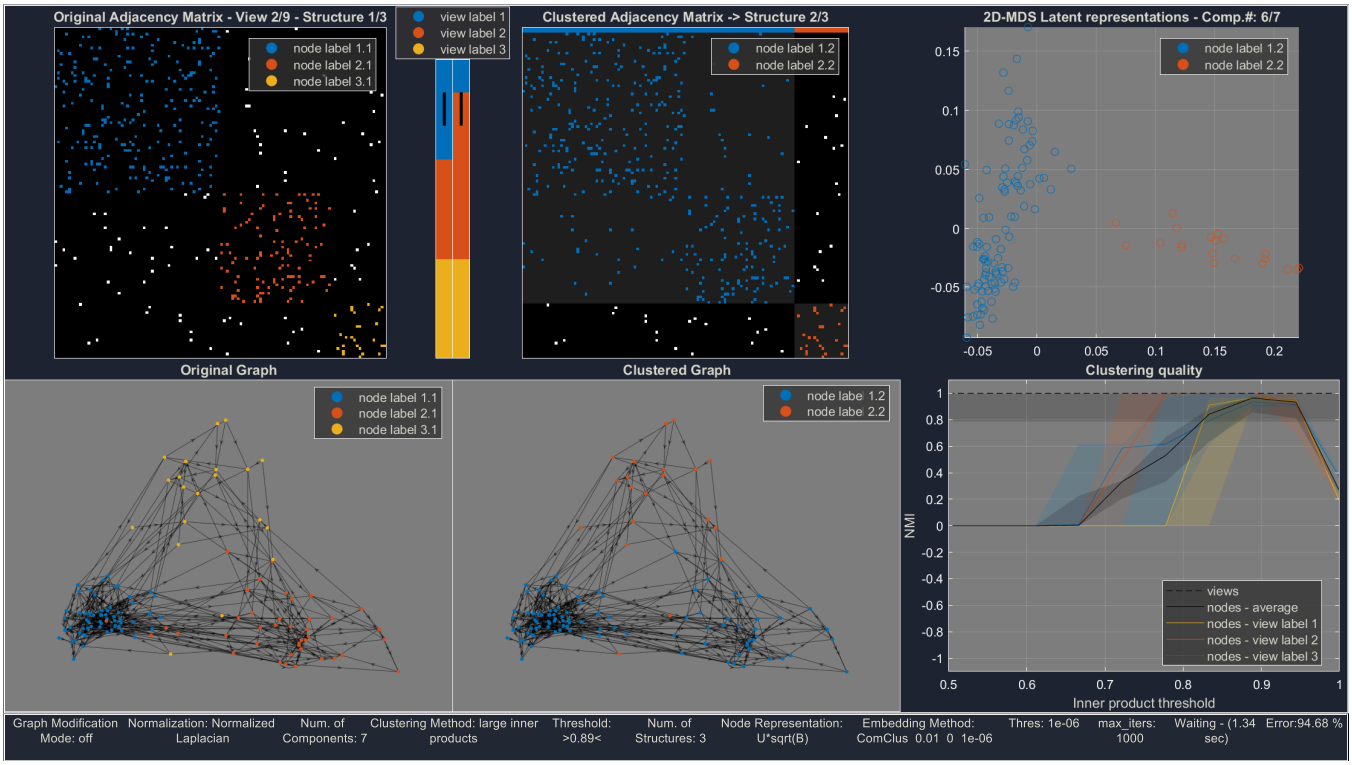


Figure 2: Single-Combination Parameter Explorer.

2.3 Multi-Graph Explorer

Here we present a flexible tool consisting of interconnected visual modules that allow the user to interactively generate and navigate multi-graphs, and explore the outputs of multi-graph methods. We separate this tool into two distinct parts: one focusing on detailed qualitative overviews of embeddings and clusterings corresponding to a particular combination of parameter values, and the other providing high-level quantitative performance comparisons between different combinations of parameter values.

2.3.1 Single-Combination Parameter Explorer. This tool provides an interactive graphical user interface as shown in Figure 2. This interface is split into the bottom part which allows the user to specify the desired parameters for the entire graph analysis workflow, and the upper part which contains the various interconnected visual modules. Our implemented graph analysis workflow currently provides interactive adjustment of the following features:

- Graph properties: Effectively a graphical user interface for the graph generator discussed in subsection 2.1.
- Graph preprocessing: Normalized Laplacian for both directed and undirected graphs.
- Embedding generation: High-quality implementations for ComClus [7], CMNC [1] and a customized Richcom [3].
- Embedding postprocessing: Various methods of modifying the embeddings before clustering them.
- Embedding clustering: k-means, maximum likelihood, and inner-product thresholding.
- Clustering quality measures: Normalized Mutual Information (NMI), Adjusted Rand Index (ARI), Adjusted Mutual Information (AMI), Macro Silhouette Coefficient, Micro Silhouette Coefficient.

As for the visual modules, the "Original Adjacency Matrix" and "Original Graph" modules visualize a specific view of the graph, where darker colors represent lower edge weights, and the different color hues represent different ground-truth labels for the corresponding nodes. Similarly, we have the "Clustered Adjacency Matrix" and "Clustered Graph" modules. However, color hues here represent the calculated labels of nodes instead of the ground-truth ones, and the adjacency matrix is permuted so that nodes assigned to the same calculated cluster are next to each other. Additionally, since the permutations can make comparisons with the "Original Adjacency Matrix" difficult, the "Clustered Adjacency Matrix" also includes a colored horizontal bar that indicates the ground-truth labels of the nodes. Next, we have the two vertical colored bars, with the one at the left indicating the ground-truth labels of the views, and the one at the right showing their calculated labels. The first view is located at the top and the last one at the bottom, and the black line indicates the view that the user is currently inspecting. Note that the color hue of each calculated node and view cluster is selected to be the same as the color hue of the ground-truth cluster that it is the most similar to. Then, the 2D multi-dimensional scaled node embeddings of the selected calculated view cluster are visualized for a more direct and intuitive analysis of their quality and clustering structure. Also, we have the "Clustering quality"

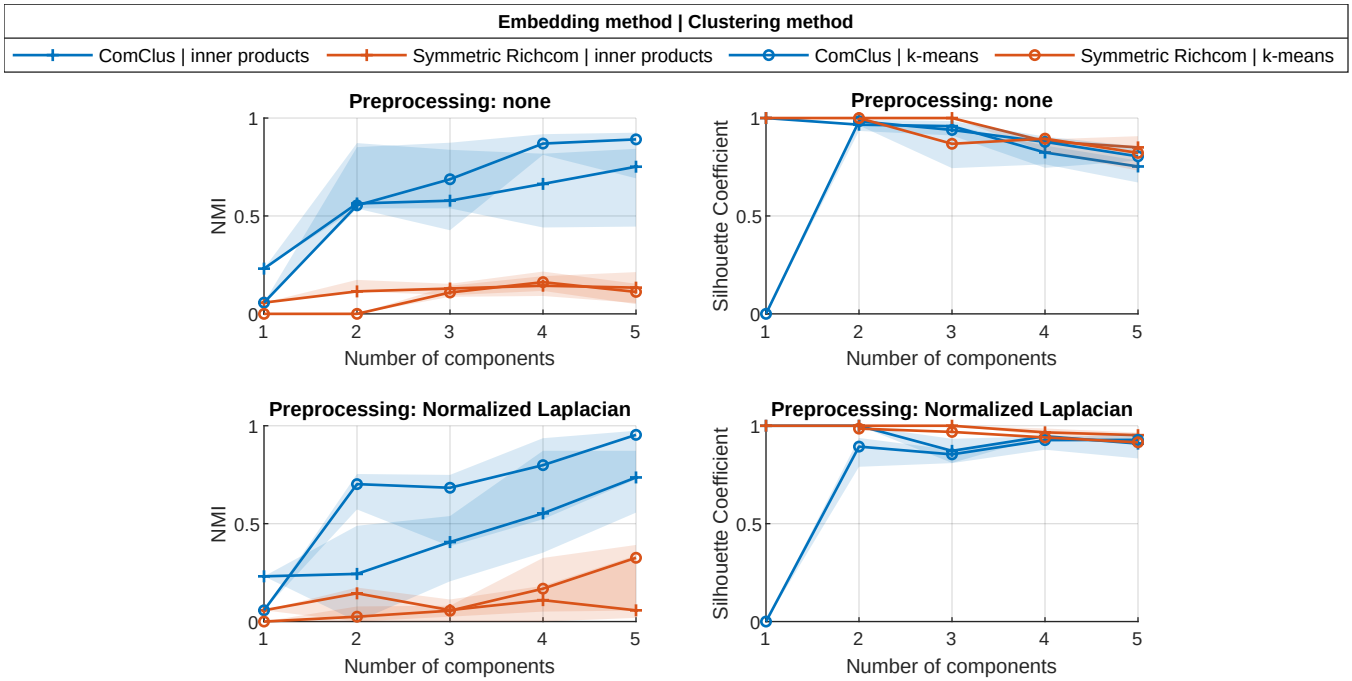


Figure 3: Multi-Combination Parameter Explorer.

module which allows the user to perform basic parameter sweeps of individual parameters. In turn, this enables the user to further fine-tune the value of any parameter via the use of quantitative clustering quality measures. Lastly, note that one of our goals is to enable users to introduce their own custom visual modules as well, and seamlessly integrate them with the existing ones.

2.3.2 Multi-Combination Parameter Explorer. Note that although the previous interface can provide very nuanced information about a graph, it is by design primarily focused on the analysis of a single combination of parameter values at a time. However, this could be an issue when a high-quality combination is not known a priori. To this end, we developed a tool that can offer a more holistic view of the results by simultaneously visualizing the clustering quality for multiple combinations of parameter values in a condensed and organized way. Specifically, each parameter can be assigned to the following visual dimensions:

- horizontal axis of each figure
- line color
- line marker
- different figures horizontally
- different figures vertically
- aggregated as median, 25-th and 75-th percentiles

Figure 3 illustrates a simple case study of how this tool enables the user to gain multiple insights into various parts of the clustering workflow. Specifically, we can see from the two NMI figures that ComClus identifies the ground-truth node clusters much better than Symmetric Richcom. However, based on the Silhouette Coefficient figures, we observe that it does not necessarily generate embeddings with a more pronounced clustering structure. Then, we observe that, with respect to NMI, k-means tends to provide better clustering performance compared to inner-products thresholding, although this advantage disappears when looking at the Silhouette Coefficient. Another useful observation is that preprocessing the graph with the Normalized Laplacian leads to better robustness against overfitting as the number of components increases, as shown in the Silhouette Coefficient plots. Lastly, note that, although, as measured by NMI, the clustering of the nodes becomes more similar to the ground-truth clustering as the number of components increases, their intrinsic clustering structure, as measured by the Silhouette Coefficient, tends to worsen.

3 Conclusions & Future Work

Due to the complexities of real-world multi-graphs and the challenges that these complexities impose on the design of relevant graph modeling methods, we proposed and developed Multi-Graph Explorer. This is a MATLAB software whose goal is to provide both researchers and practitioners with a comprehensive and efficient analysis workflow for navigating real-world multi-graph problems. In the future, we aim to further increase the modularity and augmentability of the software, along with expanding the scope of its applications. Additionally, porting the software to Python is high in our priorities as a means to increase its accessibility and its integration with existing popular machine learning workflows.

Acknowledgments

This research was supported by the National Science Foundation under CAREER grant no. IIS 2046086, grant no. IIS 1901379, and CREST Center for Multidisciplinary Research Excellence in Cyber-Physical Infrastructure Systems (MECIS) grant no. 2112650.

References

[1] Zitai Chen, Chuan Chen, Zibin Zheng, and Yi Zhu. 2019. Tensor decomposition for multilayer networks clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 3371–3378.

[2] Xiaowen Dong, Pascal Frossard, Pierre Vandergheynst, and Nikolai Nefedov. 2012. Clustering with multi-layer graphs: A spectral perspective. *IEEE Transactions on Signal Processing* 60, 11 (2012), 5820–5831.

[3] Ekta Gujral, Ravdeep Pasricha, and Evangelos Papalexakis. 2020. Beyond rank-1: Discovering rich community structure in multi-aspect graphs. In *Proceedings of The Web Conference 2020*. 452–462.

[4] Abhishek Kumar, Piyush Rai, and Hal Daume. 2011. Co-regularized multi-view spectral clustering. *Advances in neural information processing systems* 24 (2011).

[5] Xinhai Liu, Shuiwang Ji, Wolfgang Glänzel, and Bart De Moor. 2012. Multiview partitioning via tensor methods. *IEEE Transactions on Knowledge and Data Engineering* 25, 5 (2012), 1056–1069.

[6] Matteo Magnani, Obaida Hanteer, Roberto Interdonato, Luca Rossi, and Andrea Tagarelli. 2021. Community detection in multiplex networks. *ACM Computing Surveys (CSUR)* 54, 3 (2021), 1–35.

[7] Jingchao Ni, Wei Cheng, Wei Fan, and Xiang Zhang. 2017. ComClus: A self-grouping framework for multi-network clustering. *IEEE transactions on knowledge and data engineering* 30, 3 (2017), 435–448.

[8] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. 2015. A review of relational machine learning for knowledge graphs. *Proc. IEEE* 104, 1 (2015), 11–33.

[9] Rongkai Xia, Yan Pan, Lei Du, and Jian Yin. 2014. Robust multi-view spectral clustering via low-rank and sparse decomposition. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 28.

[10] Weiqiong Zhong, Haizhong An, Xiangyun Gao, and Xiaoqi Sun. 2014. The evolution of communities in the international oil trade network. *Physica A: Statistical Mechanics and its Applications* 413 (2014), 42–52.