# Modularity aided consistent attributed graph clustering via coarsening

**Samarth Bhatia*** SAMARTH.BHATIA23@ALUMNI.IITD.AC.IN *Indian Institute of Technology, Delhi*
**Yukti Makhija*** YUKTI.MAKHIJA@ALUMNI.IITD.AC.IN *Indian Institute of Technology, Delhi*
**Manoj Kumar** EEZ208646@IITD.AC.IN *Indian Institute of Technology, Delhi*
**Sandeep Kumar** KSANDEEP@EE.IITD.AC.IN *Indian Institute of Technology, Delhi*

* *denotes equal contribution*

## Abstract

Graph clustering is an important unsupervised learning technique for partitioning graphs with attributes and detecting communities. However, current methods struggle to accurately capture true community structures and intra-cluster relations, be computationally efficient, and identify smaller communities. We address these challenges by integrating coarsening and modularity maximization, effectively leveraging both adjacency and node features to enhance clustering accuracy. We propose a loss function incorporating log-determinant, smoothness, and modularity components using a block majorization-minimization technique, resulting in superior clustering outcomes. The method is theoretically consistent under the Degree-Corrected Stochastic Block Model (DC-SBM), ensuring asymptotic error-free performance and complete label recovery. Our provably convergent and time-efficient algorithm seamlessly integrates with Graph Neural Networks (GNNs) and Variational Graph AutoEncoders (VGAEs) to learn enhanced node features and deliver exceptional clustering performance. Extensive experiments on benchmark datasets demonstrate its superiority over existing state-of-the-art methods for both attributed and non-attributed graphs.

## 1. Introduction

Clustering is an unsupervised learning method that groups nodes based on their attributes or graph structure, with applications in social network analysis, genetics, bio-medicine, knowledge graphs, and computer vision. Existing graph clustering algorithms predominantly fall into cut-based, similarity-driven, or modularity-based categories.

Cut-based methods [2, 38, 41], aiming to minimize the number of edges (or similar metric) in a cut, may fall short in capturing the true community structure if the cut's edge count doesn't significantly deviate from random graph expectations. Similarity-based techniques, reliant on pairwise node similarities, group nodes with shared characteristics. Most deep learning based approaches fall under this category. These can be computationally intensive and susceptible to noise, yielding suboptimal results, especially in sparse or noisy data scenarios. Modularity-based methods rely on a statistical approach, measuring the disparity in edge density between a graph and a random graph with the same degree sequence. These may exhibit a resolution limit [10], neglecting smaller communities. Each clustering approach, whether cut-based, similarity-based, or modularity-based, bears its own set of limitations that necessitates careful consideration based on the characteristics of the underlying graph data. Refer to Appendix A for a survey on existing methods.

Some graph reduction techniques like coarsening [9, 20, 26, 27] can also facilitate clustering. In the context of graph coarsening, the objective is to learn a reduced graph by merging similar nodes

© S. Bhatia*, Y. Makhija*, M. Kumar & S. Kumar.

into supernodes. This can be extended to clustering by reducing the original graph such that each class corresponds to a supernode. However, relying solely on coarsening may lead to suboptimal performance, particularly when the order of the reduction from the original graph size to the number of classes is large and results in a significant loss of information.

In this study, we introduce an optimization-based framework that enhances clustering by leveraging both graph adjacency and node features. Our approach strategically combines coarsening and modularity maximization to refine partitioning outcomes. The algorithm minimizes a nuanced loss function, the Q-MAGC objective, which includes a $\log\det$ term, smoothness, and modularity components. This multi-block non-convex optimization problem is solved using a block majorization-minimization technique, demonstrating convergence and efficiency.

We also present the Q-GCN algorithms, which embeds our Q-MAGC objective into Graph Convolutional Networks (GCNs), improving clustering outcomes through better learned node representations. Additionally, we introduce Q-VGAE and Q-GMM-VGAE algorithms, incorporating Variational Graph Auto-Encoders (VGAEs) to further enhance clustering accuracy. Our comprehensive experiments on synthetic and real-world datasets showcase notable improvements in clustering performance, highlighting the robustness and superiority of our proposed methods.

**Key Contributions.**

- We present the first optimization-based framework for attributed graph clustering through coarsening via modularity maximization, demonstrating efficiency, theoretical convergence, and addressing limitations of existing methods. The paper offers comprehensive analysis and provides theoretical guarantees including KKT optimality, and convergence (Theorem 1), which are often absent in prior research.

- Our method is theoretically consistent under a Degree-Corrected SBM, showing asymptotically no errors (weakly consistent) and complete recovery of the original labels (strongly consistent). (Theorem 3)

- We show the seamless integration of our clustering objective with GNN-based architectures, leveraging message-passing to enhance our method (section 4). This is backed up by thorough experimental validation on diverse real-world and synthetic datasets, demonstrating superior performance compared to state-of-the-art approaches. (section 5)

- We conduct ablation studies to evaluate the behavior of the loss terms, compare runtime and complexities, and perform a comprehensive evaluation of modularity. (subsection 5.1)

**Notations.** Let $G = \{V, E, A, X\}$ be a graph with node set $V = \{v_1, v_2, ..., v_p\}$ $(|V| = p)$, edge set $E \subset V \times V\}(|E| = e)$, weight (adjacency) matrix $A$ and node feature matrix $X \in \mathbb{R}^{p\times n}$. Also, let $\mathbf{d} = A \cdot \mathbf{1}_p \in \mathbb{Z}_+^p$ be the degree vector, where $\mathbf{1}_p$ is the vector of size $p$ having all entries 1. Then, the graph Laplacian is $\Theta = \text{diag}(\mathbf{d}) - A$ and the set of all valid Laplacian matrices is defined as: $S_\Theta = \{\Theta \in \mathbb{R}^{p\times p} | \Theta_{ij} = \Theta ji \leq 0 \text{ for } i \neq j, \Theta_{ii} = \sum_{j=1}^{p} \Theta_{ij}\}$.

## 2. Background

### 2.1. Graph Coarsening

Graph coarsening is a graph dimensionality reduction technique to construct a smaller graph $G_c$ from the original graph $G = \{V, E, A, X\}$ while preserving it's properties. Graph coarsening aims to learn a mapping matrix $C \in \mathbb{R}_{+}^{p\times k}$, where $k$ is the number of nodes in the coarsened graph. Each non-zero entry of the mapping matrix $C_{ij}$ indicates that the $i$-th node of $G$ is mapped to the $j$-th supernode. Moreover, for a balanced mapping, the mapping matrix $C$ belongs to the set [20, 26, 27]: $\mathcal{C} = \left\{ C \in \mathbb{R}_{+}^{p\times k} \,|\, \langle C_i, C_j \rangle = 0 \,\forall\, i \neq j, \quad \langle C_i, C_i \rangle = d_i, \|C_i\|_0 \geq 1 \text{ and } \left\|[C^\top]_i\right\|_0 = 1 \right\}$

For $C \in \mathcal{C}$, the relationship between the original graph Laplacian $\Theta$, the coarsened graph Laplacian $\Theta_c$, and the mapping matrix $C$ is given by $\Theta_c = C^T\Theta C$.

## 2.2. Spectral Modularity Maximization

Spectral Clustering is the most direct approach to graph clustering, where we minimize the volume of inter-cluster edges. Modularity, introduced in Newman [32], is the difference between the number of edges within a cluster $\mathbf{C_i}$ and the expected number of such edges in a random graph with an identical degree sequence. It is mathematically defined as:

$$\mathcal{Q} = \frac{1}{2e}\sum_{i,j=1}^{k}\left[A_{ij} - \frac{d_i d_j}{2e}\right]\delta(c_i, c_j) \qquad \delta(c_i, c_j) = 1 \text{ if } i = j \text{ otherwise } 0 \tag{1}$$

Maximizing this form of modularity is NP-hard [5]. However, we can approximate it using a spectral relaxation, which involves a modularity matrix $B$. The modularity matrix $B$ and spectral modularity are defined as:

$$B = A - \frac{\mathbf{d}\mathbf{d}^T}{2e}, \qquad \mathbf{d} = A \cdot \mathbb{1}, \qquad \mathcal{Q} = \frac{1}{2e}Tr(C^T BC) \tag{2}$$

Modularity is closely associated with community detection. It's special spectral properties (Section B) make $B$ an ideal choice for clustering. While modularity maximization has been studied, heuristic algorithms for it are computationally intensive, such as the Newman-Girvan algorithm with $\mathcal{O}(p^3)$ time complexity. The Louvain/Leiden algorithms [4, 31] improve on this.

## 3. Proposed Method: MAGC

Current graph clustering methods often struggle to accurately capture both intra-cluster and inter-cluster relationships, achieve computational efficiency, and identify smaller communities, thereby limiting their effectiveness. These methods face significant challenges including instability and the lack of convergence guarantees, especially when employing coarsening techniques. Modularity maximization, despite extensive study, relies on computationally intensive heuristics and lacks theoretical convergence guarantees. We propose an optimization-driven framework that strategically integrates coarsening and modularity. By incorporating both adjacency and node features, our approach aims to robustly capture intra-cluster and inter-cluster dynamics. This framework ensures stability, guarantees convergence, and consistently delivers superior clustering results with enhanced computational efficiency compared to existing methods.

Given a graph $G = \{V, E, A, X\}$, to obtain the clustering matrix $C$ we formulate the following optimization problem :

$$\min_{X_C, C} \mathcal{L}_{MAGC} = \operatorname{tr}(X_C^T C^T \Theta C X_C) - \frac{\beta}{2e}\operatorname{tr}(C^T BC) - \gamma \log\det(C^T\Theta C + J)$$

$$\text{subject to } C \in \mathcal{S}_{\mathcal{C}} = \{C \in \mathbb{R}^{p\times l} \,|\, \left\|C_i^T\right\|_2^2 \leq 1\} \;\forall i, X = CX_C \;\text{ where, } J = \frac{1}{k}\mathbf{1}_{k\times k} \tag{3}$$

The term $\text{tr}(X_C^T C^T \Theta C X_C)$ represents the smoothness (Dirichlet energy) of the coarsened graph as $C^T \Theta C$ is the Laplacian of the coarsened graph. Minimizing smoothness ensures that the clusters or supernodes with similar features are linked with stronger weights. The term $\text{tr}(C^T BC)$ corresponds to the graph's modularity, enhancing the quality of the clusters formed. The term $-\log\det(C^T \Theta C + J)$ is crucial for maintaining inter-cluster edges in the coarsened graph. For a connected graph matrix with $k$ super-nodes or clusters, the rank of $C^T \Theta C$ is $k-1$. Adding We add $J$ to $C^T \Theta C$ to make a full-rank matrix without altering it's row and column space. [21].

Problem (3) is a multi-block non-convex optimization problem. All terms except modularity are convex in nature, which we prove in Appendix C. We iteratively update $C$ and $X_C$ alternately while keeping the other constant. This process continues until convergence or the stopping criteria are met. Since the constraint $X = CX_C$ is hard and difficult to enforce, we relax it by adding the term $\frac{\alpha}{2}\|X - CX_C\|_F^2$ to the objective. This term ensures that each node is assigned to a cluster, leaving no node unassigned.

**Update rule of C.** Treating $X_C$ as constant and $C$ as a variable the sub-problem for $C$ is:

$$\min_C f(C) = \text{tr}(X_C^T C^T \Theta C X_C) - \frac{\beta}{2e}\text{tr}(C^T BC) - \gamma \log\det(C^T \Theta C + J) + \frac{\alpha}{2}\|X - CX_C\|_F^2$$
$$\text{subject to } C \in \mathcal{S}_\mathcal{C} = \{C \in \mathbb{R}^{p\times l} | C \geq 0, \ \left\|C_i^T\right\|_2^2 \leq 1\} \ \forall i, \quad \text{where, } J = \frac{1}{k}\mathbf{1}_{k\times k} \tag{4}$$

By using the first-order Taylor series approximation, a majorised function for $f(C)$ at $C^t$ ($C$ after $t$ iterations) can be obtained as: $g(C|C^t) = f(C^t) + (C - C^t)\nabla f(C^t) + \frac{L}{2}\left\|C - C^t\right\|^2$ where $f(C)$ is $L-$Lipschitz continuous gradient function $L = \max(L_1, L_2, L_3, L_4)$ with $L_1, L_2, L_3, L_4$ being the Lipschitz constants of the four terms. We prove this in Appendix D. After ignoring the constant term, the majorised problem of (4) is $\min_{C\in\mathcal{S}_C} \frac{1}{2}C^T C - C^T\left(C^t - \frac{1}{L}\nabla f(C^t)\right)$

The optimal solution to this, found by using Karush–Kuhn–Tucker (KKT) optimality conditions is (Proof is deferred to the Appendix D):

$$C^{t+1} = \left(C^t - \frac{1}{L}\nabla f(C^t)\right)^+ \tag{5}$$

where, $\nabla f(C^t) = -2\gamma\Theta C^t({C^t}^T \Theta C^t + J)^{-1} + \alpha(C^t X_C - X)X_C^T + 2\Theta C^t X_C X_C^T - \frac{\beta}{e}BC^t$

**Update rule of $X_C$.** Treating $C$ fixed and $X_c$ as variable. The subproblem for updating $X_c$ is

$$\min_{\tilde{X}} f(\tilde{X}) = \text{tr}(X_C^T C^T \Theta C X_C) + \frac{\alpha}{2}\|X - CX_C\|_F^2 \tag{6}$$

The closed form solution of problem (6) can be obtained by putting the gradient of $f(\tilde{X})$ to zero.

$$X_C^{t+1} = \left(\frac{2}{\alpha}C^T \Theta C + C^T C\right)^{-1} C^T X \tag{7}$$

**Algorithm 1:** Q-MAGC Algorithm

```
Input: $G(X, \Theta), \alpha, \beta, \gamma, \lambda$
$t \leftarrow 0$;
while Stopping Criteria not met do
    Update $C^{t+1}$ as in Eqn. 5;
    Update $X_C^{t+1}$ as in Eqn. 7;
    $t \leftarrow t + 1$;
end
return $C^t, X_C^t$
```

**Convergence Analysis.**

**Theorem 1** *The sequence $\{C^{t+1}, X_C^{t+1}\}$ generated by Algorithm 1 converges to the set of Karush–Kuhn–Tucker (KKT) optimality points for Problem 3*

**Proof** The detailed proof can be found in the Appendix Section E. ■

**Complexity Analysis.** The worst-case time complexity of one epoch in Algorithm 1 is $\mathcal{O}(p^2k + pkn)$, and $k \ll p, n$. We discuss this more in Appendix Section G.

## 3.1. Consistency Analysis on Degree Corrected Stochastic Block Models (DC-SBM).

Let $\hat{y}_i$ denote the predicted cluster for node $v_i$. The assignment matrix $C$ is the one-hot encoding of $y$, such that $C_i = \text{one-hot}(y_i)$.

**Definition 2 (Strong and Weak Consistency)** *The clustering objective is defined to be strongly consistent if* $\lim_{p\to\infty} Pr[\hat{y} = y] \to 1$. *A weaker notion of consistency is defined by* $\lim_{p\to\infty} Pr\left[\frac{1}{p}\sum_{i=1}^{p} \mathbb{1}\{\hat{y} \neq y\} < \epsilon\right] \to 1 \;\forall\, \epsilon > 0.$

**Theorem 3** *Under the DC-SBM, $\mathcal{L}_{MAGC}$ is strongly consistent when $\lambda_p / \log p \to \infty$ and weakly consistent when $\lambda_p \to \infty$.*
**Proof** We use the framework of Theorem 4.1 of [45] for the proof, which is deferred to Appendix F ■

## 4. Integrating with GNNs

Our optimization framework integrates seamlessly with Graph Neural Networks (GNNs) by incorporating the objective (3) into the loss function. This integration can be minimized using gradient

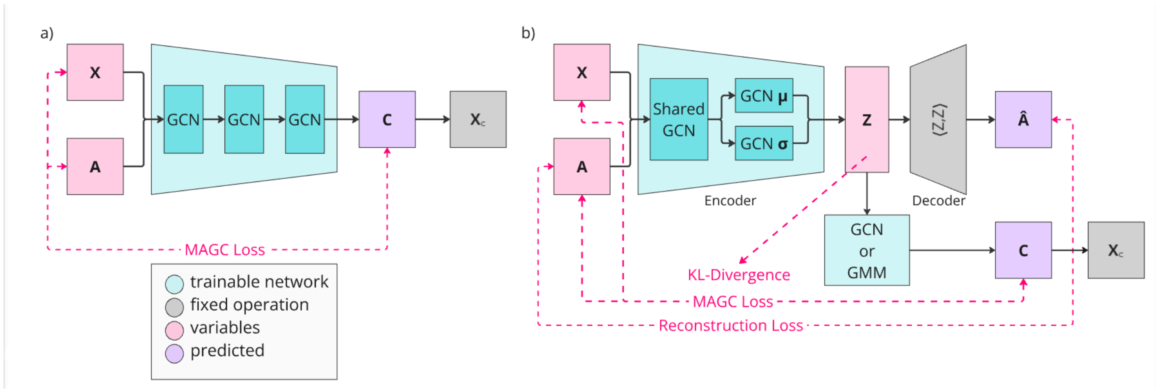


Figure 1: **a) Architecture of Q-GCN.** We want to train the encoder to learn the soft cluster assignment matrix $C$. The coarsened features $X_C$ are obtained using the relation $X_C^{t+1} = C^{t+1\dagger} X$. Finally, our proposed MAGC loss is then computed using $C$ and $X_C$.
**b) Architecture of Q-VGAE/Q-GMM-VGAE.** The three-layer GCN encoder takes $X$ and $A$ as inputs to learn the latent representation $Z$ of the graph. $Z$ is then passed through an inner-product decoder to reconstruct the adjacency matrix $\hat{A}$. The reconstruction loss is calculated between $\hat{A}$ and $A$, and the KL-divergence is applied to $Z$. In Q-VGAE (or Q-GMM-VGAE), $Z$ is also passed through a GCN layer (or GMM) to output the soft cluster assignments $C$. The MAGC loss is then computed as in Q-GCN.

descent. We demonstrate the effectiveness of this approach on several popular GNN architectures, including Graph Convolutional Networks (GCNs) [19], Variational Graph Auto-Encoders (VGAEs) [18], and a variant known as Gaussian Mixture Model VGAE (GMM-VGAE) [15]. We briefly describe this integration in Figure 1, with more details in Appendix H.

## 5. Experiments

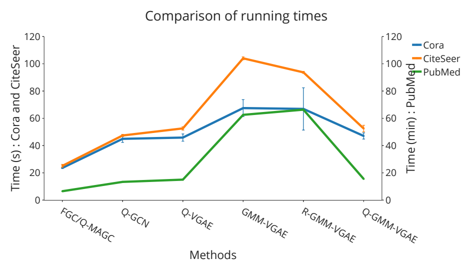
To assess the performance of our method, we compare it against three types of state-of-the-art methods: graph coarsening methods, GCN-based architectures, VGAE-based architectures and contrastive methods, and heavily modified VGAE architectures. This comprehensive evaluation allows us to demonstrate the robustness and versatility of our approach across various data and model configurations. We present our key results on the real datasets Cora, CiteSeer, and PubMed in Table 1. Our proposed method outperforms all existing methods in terms of NMI and demonstrates competitive performance in Accuracy and ARI. The best models were selected based on NMI scores. A detailed summary of all the datasets and metrics used is provided in Appendix I.

| | Cora | | | CiteSeer | | | PubMed | | |
|---|---|---|---|---|---|---|---|---|---|
| **Method** | **ACC ↑** | **NMI ↑** | **ARI ↑** | **ACC ↑** | **NMI ↑** | **ARI ↑** | **ACC ↑** | **NMI ↑** | **ARI ↑** |
| GIC [PAKDD'21] [28] | 72.5 | 53.7 | 50.8 | 69.6 | 45.3 | 46.5 | 67.3 | 31.9 | 29.1 |
| GALA [ICCV'19] [35] | **74.5** | 57.6 | 53.1 | 69.3 | 44.1 | 44.6 | 69.3 | 32.1 | **32.1** |
| DCRN [AAAI'22] [23] | 61.9 | 45.1 | 33.1 | 70.8 | 45.8 | 47.6 | 69.8 | 32.2 | 31.4 |
| FGC [JMLR'23] [20] | 53.8 | 23.2 | 20.5 | 54.2 | 31.1 | 28.2 | 67.1 | 26.6 | 27.8 |
| **Q-MAGC (Ours)** | 65.8 | 51.8 | 42.0 | 65.9 | 40.8 | 40.1 | 66.7 | **32.8** | 27.9 |
| **Q-GCN (Ours)** | 71.6 | **58.3** | **53.6** | **71.5** | **47.0** | **49.1** | 64.1 | 32.1 | 26.5 |
| SCGC [IEEE TNNLS'23] [24] | **73.8** | 56.1 | 51.7 | **71.0** | 45.2 | 46.2 | - | - | - |
| MVGRL [ICML'20] [14] | 73.2 | 56.2 | 51.9 | 68.1 | 43.2 | 43.4 | 69.3 | **34.4** | **32.3** |
| VGAE [NeurIPS'16] [18] | 64.7 | 43.4 | 37.5 | 51.9 | 24.9 | 23.8 | **69.6** | 28.6 | 31.7 |
| ARGA [IJCAI'18] [34] | 64.0 | 35.2 | **61.9** | 57.3 | 34.1 | **54.6** | 59.1 | 23.2 | 29.1 |
| **Q-VGAE (Ours)** | 72.7 | **58.6** | 49.6 | 66.1 | **47.4** | 50.2 | 64.3 | 32.6 | 28.0 |
| Mod-Aware VGAE [NN'22] [37] | 67.1 | 52.4 | 44.8 | 51.8 | 25.1 | 15.5 | - | 30.0 | 29.1 |
| GMM-VGAE [AAAI'20] [15] | 71.9 | 53.3 | 48.2 | 67.5 | 40.7 | 42.4 | 71.1 | 29.9 | 33.0 |
| R-GMM-VGAE [IEEE TKDE'22] [30] | **76.7** | 57.3 | **57.9** | 68.9 | 42.0 | 43.9 | **74.0** | 33.4 | **37.9** |
| **Q-GMM-VGAE (Ours)** | 76.2 | **58.7** | 56.3 | **72.7** | **47.4** | **48.8** | 69.0 | **34.8** | 34.0 |

Table 1: Comparison of all methods on attributed datasets.

### 5.1. Ablation Studies

We conduct ablation studies on running times, modularity comparison, parameter sensitivity, visualization of the latent space and the relative importance/evolution of the different loss terms. Figure 2 acts as a small showcase of these. They are given in detail in Appendix K.



(*a*) Comparison of running times of methods. The scale for PubMed is in minutes (right axis), whereas for Cora and CiteSeer is in seconds.

| | Cora | | | CiteSeer | | | PubMed | | |
|---|---|---|---|---|---|---|---|---|---|
| | $\mathcal{C}\downarrow$ | $\mathcal{Q}\uparrow$ | NMI $\uparrow$ | $\mathcal{C}\downarrow$ | $\mathcal{Q}\uparrow$ | NMI $\uparrow$ | $\mathcal{C}\downarrow$ | $\mathcal{Q}\uparrow$ | NMI $\uparrow$ |
| DMoN | 12.2 | **76.5** | 48.8 | 5.1 | **79.3** | 33.7 | 17.7 | **65.4** | 29.8 |
| FGC | 58.4 | 25 | 23.1 | 41.6 | 41.1 | 31 | 21.6 | 44.1 | 20.5 |
| Q-MAGC | 13.3 | 72.5 | 51.7 | 16.8 | 64.9 | 40.16 | 26 | 40.3 | 28.1 |
| Q-GCN | 13.6 | 73.3 | 58.3 | 5.8 | 74.5 | 46.7 | **8.27** | 55 | 31.5 |
| VGAE | 17.6 | 60.8 | 38.1 | 12.8 | 55.8 | 21 | 13.5 | 45.8 | 26.9 |
| Q-VGAE | **9.5** | 71.5 | **58.4** | **4.6** | 72.4 | **47.3** | 9.4 | 52.12 | **31.8** |

(*b*) Comparison of modularity and conductance at the best NMI with DMoN. Note that DMoN is optimizing only modularity, whereas we are optimizing other important terms as well, as mentioned in Eqn 3, and thus gain a lot on NMI by giving up a small amount of modularity, making us closer to the ground truth.

Figure 2

## References

[1] Aritra Bhowmick, Mert Kosan, Zexi Huang, Ambuj Singh, and Sourav Medya. Dgcluster: A neural framework for attributed graph clustering via modularity maximization, 2023.

[2] Filippo Maria Bianchi, Daniele Grattarola, and Cesare Alippi. Spectral clustering with graph neural networks for graph pooling. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

[3] Peter J. Bickel and Aiyou Chen. A nonparametric view of network models and newman–girvan and other modularities. *Proceedings of the National Academy of Sciences*, 106(50):21068–21073, 2009. doi: 10.1073/pnas.0907096106. URL https://www.pnas.org/doi/abs/10.1073/pnas.0907096106.

[4] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, oct 2008. ISSN 1742-5468. doi: 10.1088/1742-5468/2008/10/P10008. URL https://dx.doi.org/10.1088/1742-5468/2008/10/P10008.

[5] Ulrik Brandes, Daniel Delling, Marco Gaertler, R. Gorke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. On modularity clustering. *Knowledge and Data Engineering, IEEE Transactions on*, 20:172 – 188, 03 2008. doi: 10.1109/TKDE.2007.190689.

[6] Jun Jin Choong, Xin Liu, and Tsuyoshi Murata. Optimizing variational graph autoencoder for community detection with dual optimization. *Entropy*, 22(2), 2020. ISSN 1099-4300. URL https://www.mdpi.com/1099-4300/22/2/197.

[7] Ganqu Cui, Jie Zhou, Cheng Yang, and Zhiyuan Liu. Adaptive graph encoder for attributed graph embedding. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '20, page 976–985, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450379984. doi: 10.1145/3394486.3403140. URL https://doi.org/10.1145/3394486.3403140.

[8] Fnu Devvrit, Aditya Sinha, Inderjit S Dhillon, and Prateek Jain. S3GC: Scalable self-supervised graph clustering. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and

Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=ld12V3vLZ5.

[9] Inderjit S. Dhillon, Yuqiang Guan, and Brian Kulis. Weighted graph cuts without eigenvectors a multilevel approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29 (11):1944–1957, 2007. doi: 10.1109/TPAMI.2007.1115.

[10] Santo Fortunato and Marc Barthélemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences*, 104(1):36–41, 2007. doi: 10.1073/pnas.0605965104. URL https://www.pnas.org/doi/abs/10.1073/pnas.0605965104.

[11] Hongyang Gao and Shuiwang Ji. Graph u-nets. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2083–2092. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/gao19a.html.

[12] Roger Guimerà and Luís A Nunes Amaral. Functional cartography of complex metabolic networks. *Nature*, 433(7028):895–900, February 2005. ISSN 0028-0836. doi: 10.1038/nature03288.

[13] Roger Guimerà and Luís Amaral. Cartography of complex networks: Modules and universal roles. *Journal of statistical mechanics (Online)*, 2005:nihpa35573, 03 2005. doi: 10.1088/1742-5468/2005/02/P02001.

[14] Kaveh Hassani and Amir Hosein Khasahmadi. Contrastive multi-view representation learning on graphs. In *Proceedings of International Conference on Machine Learning*, pages 3451–3461. 2020.

[15] Binyuan Hui, Pengfei Zhu, and Qinghua Hu. Collaborative graph convolutional networks: Unsupervised learning meets semi-supervised learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):4215–4222, Apr. 2020. doi: 10.1609/aaai.v34i04.5843. URL https://ojs.aaai.org/index.php/AAAI/article/view/5843.

[16] Brian Karrer and M. E. J. Newman. Stochastic blockmodels and community structure in networks. *Phys. Rev. E*, 83:016107, Jan 2011. doi: 10.1103/PhysRevE.83.016107. URL https://link.aps.org/doi/10.1103/PhysRevE.83.016107.

[17] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[18] Thomas N. Kipf and Max Welling. Variational graph auto-encoders, 2016.

[19] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. arXiv, 2016. doi: 10.48550/ARXIV.1609.02907. URL https://arxiv.org/abs/1609.02907.

[20] Manoj Kumar, Anurag Sharma, and Sandeep Kumar. A unified framework for optimization-based graph coarsening. *Journal of Machine Learning Research*, 24(118):1–50, 2023. URL http://jmlr.org/papers/v24/22-1085.html.

[21] Sandeep Kumar, Jiaxi Ying, José Vinícius de M Cardoso, and Daniel P Palomar. A unified framework for structured graph learning via spectral constraints. *Journal of Machine Learning Research*, 21(22):1–60, 2020.

[22] Junhyun Lee, Inyeop Lee, and Jaewoo Kang. Self-attention graph pooling. In *Proceedings of the 36th International Conference on Machine Learning*, 09–15 Jun 2019.

[23] Yue Liu, Wenxuan Tu, Sihang Zhou, Xinwang Liu, Linxuan Song, Xihong Yang, and En Zhu. Deep graph clustering via dual correlation reduction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 7603–7611, 2022.

[24] Yue Liu, Xihong Yang, Sihang Zhou, Xinwang Liu, Siwei Wang, Ke Liang, Wenxuan Tu, and Liang Li. Simple contrastive graph clustering. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–12, 2023. doi: 10.1109/TNNLS.2023.3271871.

[25] Yue Liu, Xihong Yang, Sihang Zhou, Xinwang Liu, Zhen Wang, Ke Liang, Wenxuan Tu, Liang Li, Jingcan Duan, and Cancan Chen. Hard sample aware network for contrastive deep graph clustering. In *Proc. of AAAI*, 2023.

[26] Andreas Loukas. Graph reduction with spectral and cut guarantees. *Journal of Machine Learning Research*, 20(116):1–42, 2019. URL http://jmlr.org/papers/v20/18-680.html.

[27] Andreas Loukas and Pierre Vandergheynst. Spectrally Approximating Large Graphs with Smaller Graphs. In *Proceedings of the 35th International Conference on Machine Learning*, pages 3243–3252. PMLR, 2018.

[28] Costas Mavromatis and G. Karypis. Graph infoclust: Maximizing coarse-grain mutual information in graphs. In *PAKDD*, 2021.

[29] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. Umap: Uniform manifold approximation and projection. *Journal of Open Source Software*, 3(29):861, 2018. doi: 10.21105/joss.00861. URL https://doi.org/10.21105/joss.00861.

[30] Nairouz Mrabah, Mohamed Bouguessa, Mohamed Fawzi Touati, and Riadh Ksantini. Rethinking graph auto-encoder models for attributed graph clustering. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–15, 2022. doi: 10.1109/TKDE.2022.3220948.

[31] M. E. J. Newman. Fast algorithm for detecting community structure in networks. *Phys. Rev. E*, 69:066133, Jun 2004. doi: 10.1103/PhysRevE.69.066133. URL https://link.aps.org/doi/10.1103/PhysRevE.69.066133.

[32] Mark E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences of the United States of America*, 103 23:8577–82, 2006.

[33] Krzysztof Nowicki and Tom A. B Snijders. Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 96(455):1077–1087, 2001. doi: 10.1198/016214501753208735. URL https://doi.org/10.1198/016214501753208735.

[34] Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, Lina Yao, and Chengqi Zhang. Adversarially regularized graph autoencoder for graph embedding. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 2609–2615. International Joint Conferences on Artificial Intelligence Organization, 7 2018. doi: 10.24963/ijcai.2018/362. URL https://doi.org/10.24963/ijcai.2018/362.

[35] J. Park, M. Lee, H. Chang, K. Lee, and J. Choi. Symmetric graph convolutional autoencoder for unsupervised graph representation learning. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6518–6527, Los Alamitos, CA, USA, nov 2019. IEEE Computer Society. doi: 10.1109/ICCV.2019.00662. URL https://doi.ieeecomputersociety.org/10.1109/ICCV.2019.00662.

[36] Ketan Rajawat and Sandeep Kumar. Stochastic multidimensional scaling. *IEEE Transactions on Signal and Information Processing over Networks*, 3(2):360–375, 2017. doi: 10.1109/TSIPN.2017.2668145.

[37] Guillaume Salha-Galvan, Johannes F Lutzeyer, George Dasoulas, Romain Hennequin, and Michalis Vazirgiannis. Modularity-aware graph autoencoders for joint community detection and link prediction. *Neural Networks*, 153:474–495, 2022.

[38] Jianbo Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000. doi: 10.1109/34.868688.

[39] Anton Tsitsulin, John Palowitch, Bryan Perozzi, and Emmanuel MÃ¼ller. Graph clustering with graph neural networks. *Journal of Machine Learning Research*, 24(127):1–21, 2023. URL http://jmlr.org/papers/v24/20-998.html.

[40] Chun Wang, Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, and Chengqi Zhang. Attributed graph clustering: A deep attentional embedding approach. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 3670–3676. International Joint Conferences on Artificial Intelligence Organization, 7 2019. doi: 10.24963/ijcai.2019/509. URL https://doi.org/10.24963/ijcai.2019/509.

[41] Yen-Chuen Wei and Chung-Kuan Cheng. Towards efficient hierarchical designs by ratio cut partitioning. In *1989 IEEE International Conference on Computer-Aided Design. Digest of Technical Papers*, pages 298–301, 1989. doi: 10.1109/ICCAD.1989.76957.

[42] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/e77dbaf6759253c7c6d0efc5690369c7-Paper.pdf.

[43] Xiaotong Zhang, Han Liu, Qimai Li, and Xiao-Ming Wu. Attributed graph clustering via adaptive graph convolution. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, IJCAI'19, page 4327–4333. AAAI Press, 2019. ISBN 9780999241141.

[44] Han Zhao, Xu Yang, Zhenru Wang, Erkun Yang, and Cheng Deng. Graph debiased contrastive learning with joint representation clustering. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3434–3440. International Joint Conferences on Artificial Intelligence Organization, 8 2021. doi: 10.24963/ijcai.2021/473. URL https://doi.org/10.24963/ijcai.2021/473. Main Track.

[45] Yunpeng Zhao, Elizaveta Levina, and Ji Zhu. Consistency of community detection in networks under degree-corrected stochastic block models. *The Annals of Statistics*, 40(4):2266 – 2292, 2012. doi: 10.1214/12-AOS1036. URL https://doi.org/10.1214/12-AOS1036.

## Appendix A. Related Works

In this section, we review relevant existing works and highlight their limitations, thereby motivating the need for an improved clustering formulation.

**Graph Clustering via Coarsening.** Graph coarsening can be extended to graph clustering by reducing the size of the coarsened graph to the number of classes ($k$). However, in most graphs, the number of classes is very small, and reducing the graph to this extent may lead to poor clustering quality. DiffPool [42] learns soft cluster assignments at each layer of the GNN and optimizes two additional losses, an entropy to penalize the soft assignments and a link prediction based loss. Next, SAGPool [22] calculates attention scores and node embeddings to determine the nodes that need to be preserved or removed. Top-k [11] also works by sparsifying the graph with the learned weights. MinCutPool [2] formulates a differentiable relaxation of spectral clustering via pooling. However, Tsitsulin et al. [39] show that MinCutPool's orthogonal regularization dominates over the clustering objective and the objective is not optimized. Some disadvantages of these methods are instability and computational complexity in the case of SAGPool and DiffPool and convergence in MinCutPool. To address these challenges, we need a better loss function to perform the clustering task effectively.

**Deep Graph Clustering.** Previous literature can be classified based on contrastive and non-contrastive methods. On the non-contrastive side, Pan et al. [34] proposed ARGA and ARVGA, enforcing the latent representations to align to a prior using adversarial learning. By utilizing an attention-based graph encoder and a clustering alignment loss, Wang et al. [40] propose DAEGC. Liu et al. [23] design the DCRN model to alleviate representation collapse by a propagation regularization term minimizing the Jensen Shannon Divergence (JSD) between the latent and its product with normalized $A$. Contrastive methods include AGE [7] which builds a training set by adaptively selecting node pairs that are highly similar or dissimilar after filtering out high-frequency noises using Laplacian smoothing. Zhao et al. [44] propose GDCL to correct the sampling bias by choosing negative samples based on the clustering label.

VGAEs [18] are an increasingly popular class of GNNs that leverage variational inference [17] for learning latent graph representations in unsupervised settings. They reconstruct the adjacency matrix after passing the graph through an encoder-decoder architecture. Many attempts have been made to use VGAEs with k-means on latent embeddings, but it has been unsuitable for clustering. This is primarily because embedded manifolds obtained from VGAEs are curved and must be flattened before any clustering algorithms using Euclidean distance are applied. Refer to Appendix Section **??** for a detailed explanation. VGAEs only use a single Gaussian prior for the latent space, whereas clustering requires the integration of meta-priors. Additionally, the inner-product decoder fails to capture locality and cluster information in the formed edges. Several clustering-oriented variants of VGAEs [15, 30] have been developed that overcome most of these challenges. GMM-VGAE [15] partitions the latent space using a Gaussian Mixture Model and assigns a separate prior for each cluster to better model complex data distributions. Despite the improvement in performance, it's inner-product decoder cannot capture locality information.

**Modularity Maximization.** Various heuristic algorithms have been established that solve the NP-hard problem of modularity maximization including sampling, simulated annealing [12, 13], and greedy algorithms (Louvain/Leiden) [4, 31]. These algorithms require intensive compute and don't use node features. Modularity maximization using GNNs has also garnered attention recently. DMoN [39] optimizes only for modularity with a collapse regularization to prevent the trivial solution, but offers no theoretical guarantees about convergence. Modularity-Aware GAEs and VGAEs [37] use a prior membership matrix using Louvain algorithm and optimize for modularity using an RBF kernel as a proxy for same-community assignment. DGCLUSTER [1] is a semi-supervised method that makes use of either a subset of labels or pairwise memberships as Auxiliary Information coupled with modularity.

## Appendix B. Spectral Properties of the Modularity Matrix

$B$ is symmetric and is defined such that its row-sums and column-sums are zero, thereby making $\mathbf{1}$ one of its eigenvectors and $0$ the corresponding eigenvalue. These spectral properties of the modularity matrix are also observed in the Laplacian, as noted in Newman [32], which is a crucial element in spectral clustering. Modularity is maximized when $u_1^T s$ is maximized, where $u$ are the eigenvectors of $B$ and $s$ is the community assignment vector, i.e., placing the majority of the summation in $Q$ on the first (and largest) eigenvalue of $B$.

## Appendix C. Convexity of terms in the optimization objective (3)

When $X_C$ is kept constant, $\mathcal{L}_{MAGC}$ gets reduced to:

$$\min_C f(C) = \text{tr}(X_C^T C^T \Theta C X_C) + \frac{\alpha}{2} \|CX_C - X\|_F^2 - \frac{\beta}{2e}\text{tr}(C^T B C) - \gamma \log\det(C^T \Theta C + J) + \frac{\lambda}{2} \left\|C^T\right\|_{1,2}^2 \tag{8}$$

$$\text{subject to } \mathcal{C} \in \mathcal{S}_c(2.1) \text{ where, } J = \frac{1}{k}\mathbf{1}_{k\times k} \tag{9}$$

The term $\text{tr}(X_C^T C^T \Theta C X_C)$ is convex function in $C$. This result can be derived easily using Cholesky Decomposition on the positive semi-definite matrix $\Theta$ (i.e. $\Theta = L^T L$):

$$\text{tr}(X_C^T C^T \Theta C X_C) = \text{tr}(X_C^T C^T L^T L C X_C) = \text{tr}((LCX_C)^T LCX_C) = \|LCX_C\|_F^2 \tag{10}$$

Frobenius norm is a convex function, and the simplified expression is linear in C. Hence we can deduce that the tr(.) term is convex in C. The terms $\|CX_C - X\|_F^2$ and $\left\|C^T\right\|_{1,2}^2$ are convex because Frobenius norm and $l_{1,2}$ norm are convex in C.

For proving the convexity of $-\log\det(C^T \Theta C + J)$ we restrict function to a line. We define a function $g$:

$$g(t) = f(z + tu) \text{where, } t \in dom(g), z \in dom(f), u \in \mathbb{R}^n. \tag{11}$$

A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if $g : \mathbb{R} \to \mathbb{R}$ is convex.

The graph Laplacian matrix of the coarsened graph ($\Theta_c$) is symmetric and positive semi-definite having a rank of k-1. To convert $\Theta_c$ to positive definite matrix, we add a rank 1 matrix $J = \frac{1}{k}\mathbf{1}_{k\times k}$. ($\Theta_c + J = L^T L$)

$$f(L) = -\log\det(C^T \Theta C + J) = -\log\det(L^T L) \tag{12}$$

Now substituting L = Z + tU in the above equation.

$$\begin{aligned}
g(t) &= -\log\det((Z+tU)^T(Z+tU)) && (13)\\
&= -\log\det(Z^TZ + t(Z^TU + U^TZ)t^2U^TU) && (14)\\
&= -\log\det(Z^T(I + t(UZ^{-1} + (UZ^{-1})^T) + t^2(Z^{-1})^TU^TUZ^{-1})Z) && (15)\\
\text{substituting } P &= VZ^{-1} && (16)\\
&= -(\log\det(Z^TZ) + \log\det(I + t(P+P^T) + t^2P^TP)) && (17)\\
\text{Eigenvalue decomposition of} P &= Q\Lambda Q^T \text{and} QQ^T = I && (18)\\
&= -(\log\det(Z^TZ) + \log\det(QQ^T + 2tQ\Lambda Q^T + t^2Q\Lambda^2Q^T)) && (19)\\
&= -(\log\det(Z^TZ) + \log\det(Q(I + 2t\Lambda + t^2\Lambda^2)Q^T)) && (20)\\
&= -\log\det(Z^TZ) - \sum_{i=1}^{n}\log(1 + 2t\lambda_i + t^2\lambda^2) && (21)
\end{aligned}$$

Finding double derivative of $g(t)$:

$$g\text{''}(t) = \sum_{i=1}^{n}\frac{2\lambda_i^2(1+t\lambda_i)^2}{(1+2t\lambda_i+t^2\lambda_i^2)^2} \tag{22}$$

Since $g\text{''}(t) \geq 0 \forall\, t \in \mathbb{R}$, $g(t)$ is a convex function in $t$. This implies $f(L)$ is convex in $L$. We know that, $C^T\Theta C + J = L^TL$ so,

$$L = \Theta^{\frac{1}{2}}C + \frac{1}{\sqrt{kp}}\mathbf{1}_{p\times k} \tag{23}$$

Since $L$ is linear in $C$ and $f(L)$ is convex in $L$, $-\log\det(C^T\Theta C + J)$ is convex in C.

## Appendix D. Optimal Solution of Optimization Objective in Equation (4)

We first show that the function $f(C)$ is $L - Lipschitz$ continuous gradient function with $L = \max(L_1, L_2, L_3, L_4, L_5)$, where $L_1, L_2, L_3, L_4, andL_5$ are the Lipschitz constants of $\text{tr}(X_C^TC^T\Theta CX_C)$, $\frac{\alpha}{2}\|CX_C - X\|_F^2$, $-\frac{\beta}{2e}\text{tr}(C^TBC)$, $-\gamma\log\det(C^T\Theta C + J)$, and $\frac{\lambda}{2}\|C^T\|_{1,2}^2$.

For the $\text{tr}(X_C^TC^T\Theta CX_C)$ term, we apply triangle inequality and employ the property of the norm of the trace operator: $||tr|| = \underset{M\neq 0}{sup}\frac{|tr(M)|}{||M||_F}$.

$$|tr(X_C^TC_1^T\Theta C_1X_C) - tr(X_C^TC_2^T\Theta C_2X_C)| \tag{24}$$
$$= |tr(X_C^TC_1^T\Theta C_1X_C) - tr(X_C^TC_2^T\Theta C_1X_C) + tr(X_C^TC_2^T\Theta C_1X_C) - tr(X_C^TC_2^T\Theta C_2X_C)| \tag{25}$$
$$\leq |tr(X_C^TC_1^T\Theta C_1X_C) - tr(X_C^TC_2^T\Theta C_1X_C)| + |tr(X_C^TC_2^T\Theta C_1X_C) - tr(X_C^TC_2^T\Theta C_2X_C)| \tag{26}$$
$$\leq ||tr||||X_C^T(C_1-C_2)^T\Theta C_1X_C||_F + ||tr||||X_C^TC_2^T\Theta(C_1-C_2)X_C||_F \tag{27}$$
$$\leq ||tr||||X_C||_F||\Theta||||C_1-C_2||_F(||C_1||_F+||C_2||_F) \text{ (Frobenius Norm Property)} \tag{28}$$
$$\leq 2\sqrt{p}||tr||||X_C||_F||\Theta||||C_1-C_2||_F \;\; (||C_1||_F = ||C_2||_F = \sqrt{p}) \tag{29}$$
$$\leq L_1||C_1-C_2||_F \tag{30}$$

The second term is $\frac{\alpha}{2}\left\|CX_C - X\right\|_F^2$ can be written as:

$$\frac{\alpha}{2}tr((CX_C-X)^T(CX_C-X)) \tag{31}$$
$$= \frac{\alpha}{2}tr(X_C^TC^TCX_C - X^TCX_C + X^TX - X_C^TC^TX) \tag{32}$$
$$= \frac{\alpha}{2}(tr(X_C^TC^TCX_C) - tr(X^TCX_C) + tr(X^TX) - tr(X_C^TC^TX)) \tag{33}$$

All the terms except $tr(X^TX)$ (constant with respect to C) in obtained in the expression will follow similar proofs to tr($X_C^TC^T\Theta CX_C$).

Next we consider the modularity term:

$$|tr(C_1^TBC_1) - tr(C_2^TBC_2)| \tag{34}$$
$$= |tr(C_1^TBC_1) - tr(C_2^TBC_1) + tr(C_2^TBC_1) - tr(C_2^TBC_2)| \tag{35}$$
$$\leq |tr(C_1^TBC_1) - tr(C_2^TBC_1)| + |tr(C_2^TBC_1) - tr(C_2^TBC_2)| \tag{36}$$
$$\leq ||tr||||(C_1-C_2)^TBC_1||_F + ||tr||||(C_1-C_2)^TBC_2||_F \tag{37}$$
$$\leq ||tr||||B||||C_1-C_2||_F(||C_1||_F+||C_2||_F) \text{ (Frobenius Norm Property)} \tag{38}$$
$$\leq L_3||C_1-C_2||_F \tag{39}$$

The Lipschitz constant for $-\gamma\log\det(C^T\Theta C + J)$ is linked to the smallest non-zero eigenvalue of the coarsened Laplacian matrix ($\Theta_c$) and is bounded by $\frac{\delta}{(k-1)^2}$ [36], where $\delta$ is the minimum non-zero weight of $G_c$.

$tr(\mathbf{1}^TC^TC\mathbf{1})$ can be proved to be $L_5 - Lipschitz$ like the modularity and Dirichlet energy (smoothness) terms. This concludes the proof.

The majorized problem for L-Lipschitz and differentiable functions can now be applied. The Lagrangian of the majorized problem, (3) is:

$$\mathcal{L}(C, X_C, \mu) = \frac{1}{2}C^TC - C^TA - \mu_1^TC + \mu_2^T\left[\,\left\|C_1^T\right\|_2^2 - 1, \cdots, \left\|C_i^T\right\|_2^2 - 1, \cdots, \left\|C_p^T\right\|_2^2 - 1\right]^T \tag{40}$$

where $\mu = \mu_1 || \mu_2$ are the dual variables and $A = \Big(C - \frac{1}{L}\nabla f(C)\Big)^+$

The corresponding KKT conditions (w.r.t $C$) are:

$$C - A - \mu_1 + 2[\mu_{2_o} C_0^T, \cdots, \mu_{2i} C_i^T, \cdots, \mu_{2p} C_p^T] = 0 \tag{41}$$

$$\mu_2^T \Big[ \|C_1^T\|_2^2 - 1, \cdots, \|C_i^T\|_2^2 - 1, \cdots, \|C_p^T\|_2^2 - 1, \Big]^T = 0 \tag{42}$$

$$\mu_1^T C = 0 \tag{43}$$

$$\mu_1 \geq 0 \tag{44}$$

$$\mu_2 \geq 0 C \qquad \geq 0 \tag{45}$$

$$\big\|[C^T]_i\big\|_2^2 \leq 1 \; \forall i \tag{46}$$

The optimal solution to these KKT conditions is:

$$C = \frac{(A)^+}{\sum_i \|[A^T]_i\|_2} \tag{47}$$

## Appendix E. Proof of Theorem 1 (Convergence)

In this section, we prove that the sequence $\{C^{t+1}, X_C^{t+1}\}$ generated by Algorithm 1 converges to the set of Karush–Kuhn–Tucker (KKT) optimality points for Problem (3).

The Lagrangian of Problem (3) comes out to be:

$$\mathcal{L}(C, X_C, \mu) = \text{tr}(X_C^T C^T \Theta C X_C) + \frac{\alpha}{2} \|C X_C - X\|_F^2 - \frac{\beta}{2e} \text{tr}(C^T B C) \tag{48}$$

$$- \gamma \log \det(C^T \Theta C + J) + \frac{\lambda}{2} \|C^T\|_{1,2}^2 - \mu_1^T C + \sum_i \mu_{2i} \Big[ \|C_i^T\|_2^2 - 1 \Big] \tag{49}$$

where $\mu = \mu_1 || \mu_2$ are the dual variables.

**w.r.t.** $C$, the KKT conditions are

$$2\Theta C X_C X_C^T + \alpha (C X_C - X) X_C^T - \frac{\beta}{e} B C - 2\gamma \Theta C (C^T \Theta C + J)^{-1} \tag{50}$$

$$+ \lambda C \mathbf{1}_{k \times k} - \mu_1 + 2[\mu_{2_o} C_0^T, \cdots, \mu_{2i} C_i^T, \cdots, \mu_{2p} C_p^T] = 0$$

$$\mu_2^T \Big[ \|C_1^T\|_2^2 - 1, \cdots, \|C_i^T\|_2^2 - 1, \cdots, \|C_p^T\|_2^2 - 1 \Big]^T = 0 \tag{51}$$

$$\mu_1^T C = 0 \tag{52}$$

$$\mu_1 \geq 0 \tag{53}$$

$$\mu_2 \geq 0 \tag{54}$$

$$C \geq 0 \tag{55}$$

$$\big\|[C^T]_i\big\|_2^2 \leq 1 \; \forall i \tag{56}$$

Now, $C^\infty \equiv \lim_{t \to \infty} C^t$ is found from (5) as:

$$C^{\infty} = C^{\infty} + \frac{1}{L}\Bigg(2\Theta C^{\infty} X_C^{\infty} X_C^{\infty} + \alpha(C^{\infty} X_C - X) X_C^{\infty} - \frac{\beta}{e} B C^{\infty} \tag{57}$$

$$- 2\gamma \Theta C^{\infty} ({C^{\infty}}^T \Theta C^{\infty} + J)^{-1} + \lambda C^{\infty} \mathbf{1}_{k\times k}\Bigg)$$

$$0 = 2\Theta C^{\infty} X_C^{\infty} X_C^{\infty} + \alpha(C^{\infty} X_C - X) X_C^{\infty} - \frac{\beta}{e} B C^{\infty} \tag{58}$$

$$- 2\gamma \Theta C^{\infty} ({C^{\infty}}^T \Theta C^{\infty} + J)^{-1} + \lambda C^{\infty} \mathbf{1}_{k\times k}$$

So, for $\mu = 0$, $C^{\infty}$ satisfies the KKT conditions.
**w.r.t.** $X_C$, the KKT conditions are:

$$2C^T \Theta C X_C + \alpha C^T (C X_C - X) = 0 \tag{59}$$

So, $X^{\infty} \equiv \lim_{t\to\infty} X^t$ found from (7) will satisfy this as that equation is just a rearrangement of the KKT condition.

## Appendix F. Proof (continued) of Theorem 3 (conditions for consistency)

As previously defined in Theorem 3 $O = C^T A C$ and

$$\frac{1}{\mu_p}\mathbb{E}[O|C,t] = H(S)$$

We need to find "population version" of the loss function in terms of $H(S)$.

$$\mathbb{E}[O|C,t] = \mathbb{E}[C^T A C|C,t] = \mu_p H(S) \tag{60}$$

$$\mathbb{E}[C^T D C|C,t] = \mathbb{E}[C^T \text{diag}(\sum_{j=1}^{k} A_{ij}) C|C,t] = \mathbb{E}[\text{diag}(\sum_{j=1}^{k} O_{ij})|C,t] \tag{61}$$

$$= \mu_p \text{diag}(\sum_{j=1}^{k} H_{ij}) \tag{62}$$

So, for tr$(X_C^T C^T \Theta C X_C)$

$$\begin{aligned}\mathbb{E}[\text{tr}(X_C^T C^T \Theta C X_C)|C,t] \quad &= \text{tr}(\mathbb{E}[X_C^T C^T \Theta C X_C|C,t])\\ = \text{tr}(X_C^T \mathbb{E}[C^T \Theta C|C,t] X_C) &= \text{tr}(X_C^T \mathbb{E}[C^T D C - C^T A C|C,t] X_C)\\ = \text{tr}(X_C^T[\mu_p \text{diag}(\sum_{j=1}^{k} H_{ij}) \quad &- \mu_p H(S)] X_C)\\ = \mu_p \text{tr}(X_C^T[\text{diag}(\sum_{j=1}^{k} H_{ij}) \quad &- H(S)] X_C)\end{aligned} \tag{63}$$

Next, we have $\frac{1}{2e}\text{tr}(C^TBC)$, which has already been solved in the paper [45] in their Appendix (Page 25 of the full document).

$$\frac{1}{2e}\mathbb{E}[\text{tr}(C^TBC)] = \sum_k \left(\frac{H_{kk}}{\tilde{P}_0} - \left(\frac{H_k}{\tilde{P}_0}\right)^2\right)$$

For $\log\det(C^T\Theta C + J)$, where $J = \frac{1}{k}\mathbb{1}_{k\times k}$,

We can write $\log(\det(C^T\Theta C + J)) = \text{tr}(\log(C^T\Theta C + J))$ since, $\det(A) = e^{\text{tr}(\log(A))}$.

$$Z = C^T\Theta C + J = V\Lambda V^{-1} \tag{64}$$

$$\text{So,} \tag{65}$$

$$\text{tr}(\log(V\Lambda V^{-1})) = \text{tr}(V\log(\Lambda)V^{-1}) \tag{66}$$

$$\log\Lambda = \log(bI) + \log\left(I + \frac{\Lambda}{b} - I\right) \tag{67}$$

Using the first-order Taylor expansion of $\log(I + X) = X$, we need to choose $b$ such that

$$l = \left\|\frac{\Lambda}{b} - I\right\|_F < 1 \tag{68}$$

And for the expansion to be a good approximation, we need $l \to 0$. We will enforce this later in (78).

$$\text{tr}(V\bigg(\log(bI) + \log\left(I + \frac{\Lambda}{b} - I\right)\bigg)V^{-1}) \tag{69}$$

$$= \text{tr}(V\bigg(\log(bI) + \left(I + \frac{\Lambda}{b} - I\right)\bigg)V^{-1}) \tag{70}$$

$$= \text{tr}(V\log(bI)V^{-1}) + \frac{1}{b}\text{tr}(V\Lambda V^{-1}) - \text{tr}(I) \tag{71}$$

$$= \text{tr}(\log(bI)) + \frac{1}{b}\text{tr}(Z) - \text{tr}(I) \tag{72}$$

$$= k\log(b) + \frac{1}{b}\text{tr}(Z) - k \tag{73}$$

Finding the expectation of from (67),

$$= k\log(b) + \mathbb{E}\left[\frac{1}{b}\text{tr}(Z)\middle|\, c, t\right] - k \tag{74}$$

$$= k\log(b) + \frac{1}{b}\text{tr}(\mathbb{E}[Z|\, c, t]) - k \tag{75}$$

$$= k\log(b) + \frac{1}{b}\text{tr}(\mathbb{E}[C^T\Theta C + J|\, c, t]) - k \tag{76}$$

Using $\Theta = D - A$, (60) and (62),

$$= k\log(b) + \frac{1}{b}\text{tr}\bigg(\mu_p\text{diag}(\sum_{j=1}^{k} H_{ij}) - \mu_p H(S)\bigg) - k \tag{77}$$

which is a linear function in $H(S)$.

For the approximation to be good, we can now simplify $l$ as defined in (68):

$$l = \left\| \frac{\Lambda}{b} - I \right\|_F \tag{78}$$

$$l = \left\| \begin{matrix} \frac{1}{b}\Lambda_{11} - 1 & \frac{1}{b}\Lambda_{12} & \cdots & \frac{1}{b}\Lambda_{1k} \\ \frac{1}{b}\Lambda_{21} & \frac{1}{b}\Lambda_{22} - 1 & \cdots & \frac{1}{b}\Lambda_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{b}\Lambda_{k1} & \frac{1}{b}\Lambda_{k2} & \cdots & \frac{1}{b}\Lambda_{kk} - 1 \end{matrix} \right\|_F \tag{79}$$

Writing out this norm, we get a quadratic expression in $\frac{1}{b}$:

$$l = (\sum_{i=1}^{k}\sum_{j=1}^{k}\lambda_{ij}^2)\frac{1}{b^2} - 2(\sum_{u=1}^{k}\lambda_{ii})\frac{1}{b} + k \tag{80}$$

$$\text{or concisely, } l = \|\Lambda\|_F^2 \frac{1}{b^2} - 2\text{tr}(\Lambda)\frac{1}{b} + k \tag{81}$$

$$\text{Since, } \Lambda \text{ is a diagonal matrix,} \tag{82}$$

$$\text{tr}(\Lambda) = \sum_i \lambda_i = \text{tr}(Z) \tag{83}$$

$$\text{Also, since } \Lambda^2 \text{ is the eigenvalue matrix for } Z^2, \tag{84}$$

$$\|\Lambda\|_F^2 = \sum_i \lambda_i^2 = \text{tr}(Z^2) \tag{85}$$

$$l = \text{tr}(Z^2)\frac{1}{b^2} - 2\text{tr}(Z)\frac{1}{b} + k \tag{86}$$

Since, $l < 1 \implies l - 1 < 0 \implies l - 1 = 0$ has 2 real roots. Using simple quadratic analysis (in $\frac{1}{b}$), the discriminant $\Delta$ should be positive.

$$\Delta = 4\text{tr}(Z)^2 \; - 4(k-1)\text{tr}(Z^2) > 0 \tag{87}$$

$$\frac{\text{tr}(Z)^2}{\text{tr}(Z^2)} > k - 1 \tag{88}$$

$$\text{The minimum value of } l := l_{min} \text{ occurs at} \tag{89}$$

$$b = \frac{\text{tr}(Z^2)}{\text{tr}(Z)} = \text{tr}(Z) \cdot \frac{\text{tr}(Z^2)}{\text{tr}(Z)^2} \tag{90}$$

$$l_{min} < 1 \text{ will exist when } \frac{\text{tr}(Z)^2}{\text{tr}(Z^2)} > k - 1 \tag{91}$$

$$\text{which holds for } b < \frac{\text{tr}(Z)}{k-1} \tag{92}$$

$$\text{So we can always choose } b < \frac{2k-1}{k-1} \tag{93}$$

$$\text{since } \min \text{tr}(Z) = 2k - 1[\text{proved in } 95] \tag{94}$$

$$\begin{aligned}\min \text{tr}(C^T\Theta C + J) &= \min \text{tr}(C^T\Theta C) + \text{tr}(J) &(95)\\ &= \min \text{tr}(C^TDC) - \text{tr}(C^TAC) + 1 &(96)\\ &= 2e - 2(e-(k-1)) + 1 &(97)\\ &= 2k-1 &(98)\end{aligned}$$

Additionally, define $\tilde{\pi}_a = \sum_u x_u \Pi_{au}$ with $\sum_a \tilde{\pi}_a = 1$, since $\mathbb{E}[t_i] = 1$.

### F.1. Required Condition a): Lipschitz Continuity

Condition 1: We need to show that $|F(S_1) - F(S_2)| \leq \alpha \|S_1 - S_2\|$ (Lipschitz)

$$|F(S_1) - F(S_2)| \leq |f_1(S_1) - f_1(S_2)| + |f_2(S_1) - f_2(S_2)| + |f_3(S_1) - f_3(S_2)| \quad (99)$$

Let's first find $\|H(S_1) - H(S_2)\|_F$

$$\begin{aligned}H(S) &= (Sx)P(Sx)^T &(100)\\ \|H(S_1) - H(S_2)\|_F &= \|(S_1x)P(S_1x)^T - (S_2x)P(S_2x)^T\|_F &(101)\\ &= \|(S_1x)P((S_1-S_2)x)^T + ((S_1-S_2)x)P(S_2x)^T\|_F &(102)\end{aligned}$$

Next, we see $\|\text{diag}(\sum_{j=1}^k H(S_1)_{ij}) - \text{diag}(\sum_{j=1}^k H(S_2)_{ij})\|_F = \|\text{diag}(\sum_{j=1}^k (H(S_1)_{ij} - H(S_2)_{ij}))\|_F$

For the first term, $|f_1(H(S_1)) - f_1(H(S_2))|$, define $H(S)_i = \sum_{j=1}^k H(S)_{ij}$

$$= |\mu_p \text{tr}(X_C^T[H(S_1) - \text{diag}(\sum_{j=1}^k H(S_1)_{ij})]X_C) - \mu_p \text{tr}(X_C^T[H(S_2) - \text{diag}(\sum_{j=1}^k H(S_2)_{ij})]X_C)| \quad (103)$$

$$= |\mu_p \Big( \text{tr}(X_C^T[H(S_1) - H(S_2)]X_C) - \text{tr}(X_C^T \text{diag}([H(S_1)_i - H(S_2)_i])X_C) \Big)| \quad (104)$$

$$\leq |\mu_p| \Big( \Big|\text{tr}(X_C^T[H(S_1) - H(S_2)]X_C)\Big| - \Big|\text{tr}(X_C^T \text{diag}([H(S_1)_i - H(S_2)_i])X_C)\Big| \Big) \quad (105)$$

$$\leq |\mu_p| \Big( \|\text{tr}\| \Big\| X_C^T[H(S_1) - H(S_2)]X_C \Big\|_F - \|\text{tr}\| \Big\| X_C^T \text{diag}([H(S_1)_i - H(S_2)_i])X_C \Big\|_F \Big) \quad (106)$$

$$\leq |\mu_p| \Big( \|\text{tr}\| \|X_C\|^2 \Big\| H(S_1) - H(S_2) \Big\|_F - \|\text{tr}\| \|X_C\|^2 \Big\| \text{diag}([H(S_1)_i - H(S_2)_i]) \Big\|_F \Big) \quad (107)$$

Taking the first sub-term,

$$\begin{aligned}
&|\mu_p|\ \|\text{tr}\|\ \|X_C\|^2 \left\| H(S_1) - H(S_2) \right\|_F && (108)\\
&= \|\text{tr}\|\ \|X_C\|^2 \left\| (S_1x)P((S_1 - S_2)x)^T + ((S_1 - S_2)x)P(S_2x)^T \right\|_F && (109)\\
&\leq \|\text{tr}\|\ \|X_C\|^2\ \|P\|_F\ (\|S_1x\|_F + \|S_2x\|_F)\ \|(S_1 - S_2)x\|_F && (110)\\
&\leq \|\text{tr}\|\ \|X_C\|^2\ \|P\|_F\ (\|S_1x\|_F + \|S_2x\|_F)\ \|t\|_F\ \|(S_1 - S_2)\|_F && (111)\\
&= \alpha_1 \|(S_1 - S_2)\|_F && (112)
\end{aligned}$$

For the second sub-term, define $\tilde{S}_{ka} = \sum_u x_u S_{kau} = (Sx)_{ka}$

$$\begin{aligned}
H(S)_i = \sum_{j=1}^{k} H(S)_{ij} &= \sum_{as} \tilde{\pi}_s P_{as} \tilde{S}_{ia} = \sum_{asu} \tilde{\pi}_s P_{as} x_u S_{iau} && (113)\\
|H(S_1)_i - H(S_2)_i| &= |\sum_{asu} \tilde{\pi}_s P_{as} x_u S_{1iau} - \sum_{asu} \tilde{\pi}_s P_{as} x_u S_{2iau}| && (114)\\
&= |\sum_{asu} \tilde{\pi}_s P_{as} x_u (S_1 - S_2)_{iau}| && (115)\\
&\leq \sum_{asu} |\tilde{\pi}_s P_{as} x_u (S_1 - S_2)_{iau}| && (116)\\
&\leq \sum_{asu} |\tilde{\pi}_s P_{as} x_u| \cdot |(S_1 - S_2)_{iau}| && (117)\\
&\leq \sum_{asu} |\tilde{\pi}_s P_{as} x_u| \cdot \sum_{au} |(S_1 - S_2)_{iau}| && (118)\\
&= \alpha_1' \sum_{au} |(S_1 - S_2)_{iau}| && (119)\\
\text{So, } \|\text{diag}([H(S_1)_i - H(S_2)_i])\|_F &= \sqrt{\sum_{i=1}^{k} \left( \sum_{j=1}^{k} H(S_1)_{ij} - H(S_2)_{ij} \right)^2} && (120)\\
&\leq \sqrt{\sum_{i=1}^{k} \left( \alpha_1' \sum_{au} |(S_1 - S_2)_{iau}| \right)^2} && (121)\\
&= \alpha_1' \|S_1 - S_2\|_{1,1,2} && (122)
\end{aligned}$$

For the second term, $|f_2(H(S_1)) - f_2(H(S_2))|$

$$= \frac{\mu_p}{b}\Big|\text{tr}(H(S_1) - \text{diag}([H(S_1)_i])) - \text{tr}(H(S_2) - \text{diag}([H(S_2)_i]))\Big| \tag{123}$$

$$= \frac{\mu_p}{b}\Big|\text{tr}(H(S_1) - H(S_2)) \qquad - \text{tr}\Big(\text{diag}([H(S_1)_i - H(S_2)_i])\Big)\Big| \tag{124}$$

$$\leq \frac{\mu_p}{b}\Big(\Big|\text{tr}(H(S_1) - H(S_2))\Big| \qquad + \Big|\text{tr}\Big(\text{diag}([H(S_1)_i - H(S_2)_i])\Big)\Big|\Big) \tag{125}$$

$$\leq \frac{\mu_p}{b}\Big(\|\text{tr}\| \Big\|H(S_1) - H(S_2)\Big\|_F \qquad + \|\text{tr}\|\Big\|\text{diag}([H(S_1)_i - H(S_2)_i])\Big\|_F\Big|\Big) \tag{126}$$

As shown above, (127)

$$\leq \alpha_2\|(S_1 - S_2)\|_F \qquad + \alpha_2'\|(S_1 - S_2)\|_{1,1,2} \tag{128}$$

For the third term, it has already been proven in [45], but we also prove it here:
$|f_3(H(S_1)) - f_3(H(S_2))|$

$$= \left|\frac{\text{tr}(H(S_1))}{\tilde{P}_0} - \frac{\sum_{i=1}^k(\sum_{j=1}^k H(S_1)_{ij})^2}{\tilde{P}_0^{\,2}} - \frac{\text{tr}(H(S_2))}{\tilde{P}_0} + \frac{\sum_{i=1}^k(\sum_{j=1}^k H(S_2)_{ij})^2}{\tilde{P}_0^{\,2}}\right| \tag{129}$$

$$= \left|\frac{\text{tr}(H(S_1) - H(S_2))}{\tilde{P}_0} - \frac{\sum_{i=1}^k \left[H(S_1)_i^2 - H(S_2)_i^2\right]}{\tilde{P}_0^{\,2}}\right| \tag{130}$$

$$\leq \frac{1}{\tilde{P}_0}\Big|\text{tr}(H(S_1) - H(S_2))\Big| + \frac{1}{\tilde{P}_0^{\,2}}\Big|\sum_{i=1}^k \left[H(S_1)_i^2 - H(S_2)_i^2\right]\Big| \tag{131}$$

As shown above, (132)

$$\leq \alpha_3\|(S_1 - S_2)\|_F + \frac{1}{\tilde{P}_0^{\,2}}\Big|\sum_{i=1}^k \left[H(S_1)_i - H(S_2)_i\right] \cdot \left[H(S_1)_i + H(S_2)_i\right]\Big| \tag{133}$$

$$\leq \alpha_3\|(S_1 - S_2)\|_F + \frac{1}{\tilde{P}_0^{\,2}}\sum_{i=1}^k \left|H(S_1)_i + H(S_2)_i\right| \cdot \sum_{i=1}^k \left|H(S_1)_i - H(S_2)_i\right| \tag{134}$$

$$= \alpha_3\|(S_1 - S_2)\|_F + \alpha_3'\sum_{iau}\left|S_1 - S_2\right|_{iau} \tag{135}$$

$$= \alpha_3\|(S_1 - S_2)\|_F + \alpha_3'\|S_1 - S_2\|_{1,1,1} \tag{136}$$

### F.2. Required Condition b): Continuity of directional second derivative

Condition 2: $W = H(\mathbb{D})$

$$\frac{\partial^2}{\partial\varepsilon^2}F(M_0 + \varepsilon(M_1 - M_0), \mathbf{t_0} + \varepsilon(\mathbf{t_1} - \mathbf{t_0}))\Big|_{\varepsilon=0^+} \tag{137}$$

$$= \frac{\partial^2}{\partial\varepsilon^2}f_1(M_0 + \varepsilon(M_1 - M_0)) + \frac{\partial^2}{\partial\varepsilon^2}f_2(M_0 + \varepsilon(M_1 - M_0)) + \frac{\partial^2}{\partial\varepsilon^2}f_3(M_0 + \varepsilon(M_1 - M_0))\Big|_{\varepsilon=0^+} \tag{138}$$

Finding the directional derivative for $f_1$,

$$f_1(M_0+\varepsilon(M_1-M_0)) = \mu_p \text{tr}(X_C^T[M_0+\varepsilon(M_1-M_0) - \text{diag}(\sum_{j=1}^{k}\big(M_0+\varepsilon(M_1-M_0)\big)_{ij})]X_C) \tag{139}$$

$$= \mu_p \Bigg( \text{tr}(X_C^T M_0 X_C) + \varepsilon\, \text{tr}(X_C^T(M_1-M_0)X_C) \tag{140}$$

$$- \text{tr}(X_C^T \text{diag}([\sum_{j=1}^{k}(M_0)_{ij}])X_C) - \varepsilon\, \text{tr}(X_C^T \text{diag}([\sum_{j=1}^{k}(M_1-M_0)_{ij}])X_C)\Bigg) \tag{141}$$

$$\frac{\partial^2}{\partial\varepsilon} f_1(M_0+\varepsilon(M_1-M_0)) = \mu_p \Bigg( \text{tr}(X_C^T(M_1-M_0)X_C) - \text{tr}(X_C^T \text{diag}([\sum_{j=1}^{k}(M_1-M_0)_{ij}])X_C)\Bigg) \tag{142}$$

$$\frac{\partial^2}{\partial\epsilon^2} f_1(M_0+\varepsilon(M_1-M_0) = 0 \tag{143}$$

Finding the directional derivative for $f_2$,

$$f_2(M_0+\varepsilon(M_1-M_0)) \tag{144}$$

$$= \frac{\mu_p}{b}\text{tr}\Bigg(M_0+\varepsilon(M_1-M_0) - \text{diag}(\sum_{j=1}^{k}\big(M_0+\varepsilon(M_1-M_0)\big)_{ij})\Bigg) - \frac{1}{b} + k - k\log b \tag{145}$$

$$\frac{\partial}{\partial\varepsilon} f_2(M_0+\varepsilon(M_1-M_0)) \quad = \frac{\mu_p}{b}\text{tr}(M_1-M_0) - \frac{\mu_p}{b}\text{tr}\Bigg(\text{diag}(\sum_{j=1}^{k}(M_1-M_0)_{ij})\Bigg) \tag{146}$$

$$\frac{\partial^2}{\partial\varepsilon^2} f_2(M_0+\varepsilon(M_1-M_0)) \quad = 0 \tag{147}$$

Finding the directional derivative for $f_3$,

$$f_3(M_0+\varepsilon(M_1-M_0)) = \frac{\text{tr}(M_0+\varepsilon(M_1-M_0))}{\tilde{P_0}} - \frac{\sum_{i=1}^{k}(\sum_{j=1}^{k}(M_0+\varepsilon(M_1-M_0))_{ij})^2}{\tilde{P_0}^2} \tag{148}$$

$$= \frac{\text{tr}(M_0+\varepsilon(M_1-M_0))}{\tilde{P_0}} - \frac{\sum_{i=1}^{k}(\sum_{asu}\tilde{\pi}_s P_{as} x_u (M_0+\varepsilon(M_1-M_0))_{iau})^2}{\tilde{P_0}^2} \tag{149}$$

$$\frac{\partial}{\partial\varepsilon} f_3(M_0+\varepsilon(M_1-M_0)) =$$

$$\frac{\text{tr}(M_1-M_0)}{\tilde{P_0}} + \sum_{i=1}^{k} 2(\sum_{asu}\tilde{\pi}_s P_{as} x_u (M_1-M_0)_{iau}) \times \frac{\sum_{i=1}^{k}(\sum_{asu}\tilde{\pi}_s P_{as} x_u (M_0+\varepsilon(M_1-M_0))_{iau})^2}{\tilde{P_0}^2} \tag{150}$$

$$\frac{\partial^2}{\partial\varepsilon^2} f_3(M_0+\varepsilon(M_1-M_0)) = \frac{\sum_{i=1}^{k} 2(\sum_{asu}\tilde{\pi}_s P_{as} x_u (M_1-M_0)_{iau})^2}{\tilde{P_0}^2} \tag{151}$$

Adding up these three,

$$\frac{\partial^2}{\partial\varepsilon^2}F(M_0+\varepsilon(M_1-M_0),\mathbf{t_0}+\varepsilon(\mathbf{t_1}-\mathbf{t_0}))=\frac{\sum_{i=1}^{k}2(\sum_{asu}\tilde{\pi}_s P_{as}x_u(M_1-M_0)_{iau})^2}{\tilde{P_0}^2}$$

which is continuous in $(M_1,\mathbf{t_1})$ for all $(M_0,\mathbf{t_0})$ in a neighborhood of $(W,\pi)$.

## F.3. Required Condition c): Upper bound of first derivative

With $G(S)=F(H(S),h(S))$, $\frac{\partial G((1-\varepsilon)\mathbb{D}+\varepsilon S)}{\partial\varepsilon}|_{\varepsilon=0^+}<-C<0\;\forall\,\pi,P$

$G(S)=f_1(H(S))+f_2(H(S))+f_3(H(S))$

Let $\bar{S}=((S-\mathbb{D})t)P(\mathbb{D}t)^T+(\mathbb{D}t)P(S-\mathbb{D}t)^T$

For $f_1$

$$f_1(H((1-\varepsilon)\mathbb{D}+\varepsilon S))=\mu_p\text{tr}(X_C^T[H((1-\varepsilon)\mathbb{D}+\varepsilon S)-\text{diag}(\sum_{j=1}^{k}H((1-\varepsilon)\mathbb{D}+\varepsilon S)_{ij})]X_C) \tag{152}$$

$$H(S)=(Sx)P(Sx)^T \tag{153}$$

$$H((1-\varepsilon)\mathbb{D}+\varepsilon S)=((1-\varepsilon)\mathbb{D}x+\varepsilon St)P((1-\varepsilon)\mathbb{D}x+\varepsilon St)^T \tag{154}$$

$$=(\mathbb{D}x+\varepsilon(S-\mathbb{D})x)P(\mathbb{D}x+\varepsilon(S-\mathbb{D})x)^T \tag{155}$$

$$=(\mathbb{D}x)P(\mathbb{D}x)^T+\varepsilon(\mathbb{D}x)P((S-\mathbb{D})x)^T \tag{156}$$

$$+\varepsilon((S-\mathbb{D})x)P(\mathbb{D}x)^T+\varepsilon^2((S-\mathbb{D})x)P((S-\mathbb{D})x)^T \tag{157}$$

Finally, $$f_1((1-\varepsilon)\mathbb{D}+\varepsilon S)=\text{tr}\big(X_C^T(\mathbb{D}x)P(\mathbb{D}x)^TX_C\big)+\varepsilon\,\text{tr}\big(X_C^T(\mathbb{D}x)P((S-\mathbb{D})x)^TX_C\big) \tag{158}$$

$$+\varepsilon\,\text{tr}\big(X_C^T((S-\mathbb{D})x)P(\mathbb{D}x)^TX_C\big)+\varepsilon^2\,\text{tr}\big(X_C^T((S-\mathbb{D})x)P((S-\mathbb{D})x)^TX_C\big) \tag{159}$$

$$+\text{tr}\big(X_C^T\text{diag}([H(\mathbb{D})_i])X_C\big)+\varepsilon^2\,\text{tr}\big(X_C^T\text{diag}([H(S-\mathbb{D})_i])X_C\big) \tag{160}$$

$$+\varepsilon\,\text{tr}\big(X_C^T\text{diag}([\big(((S-\mathbb{D})x)P(\mathbb{D}x)^T+(\mathbb{D}x)P(S-\mathbb{D}x)^T\big)_i])X_C\big) \tag{161}$$

$$\text{Now,}\left.\frac{\partial f_1}{\partial\varepsilon}\right|_{\varepsilon=0^+}=\text{tr}\big(X_C^T(\bar{S}-\text{diag}([\bar{S}_i]))X_C\big) \tag{162}$$

For $f_2$,

$$f_2(H((1-\varepsilon)\mathbb{D}+\varepsilon S))=\frac{\mu_p}{b}\text{tr}\bigg(H((1-\varepsilon)\mathbb{D}+\varepsilon S)-\text{diag}([H((1-\varepsilon)\mathbb{D}+\varepsilon S)_i])\bigg)-\frac{1}{b}+k-k\log b \tag{163}$$

$$\left.\frac{\partial f_2}{\partial\varepsilon}\right|_{\varepsilon=0^+}=\frac{\mu_p}{b}\text{tr}\bigg(\bar{S}-\text{diag}([\bar{S}_i])\bigg) \tag{164}$$

For $f_3$,

$$f_3(H((1-\varepsilon)\mathbb{D}+\varepsilon S)) = \frac{1}{\tilde{P}_0}\text{tr}\Big((\mathbb{D}x)P(\mathbb{D}x)^T + \varepsilon(\mathbb{D}x)P((S-\mathbb{D})x)^T \tag{165}$$

$$+\,\varepsilon((S-\mathbb{D})x)P(\mathbb{D}x)^T + \varepsilon^2((S-\mathbb{D})x)P((S-\mathbb{D})x)^T\Big) \tag{166}$$

$$-\frac{1}{\tilde{P}_0^2}\sum_{i=1}^{k}\sum_{j=1}^{k}\Big(\Big[(\mathbb{D}x)P(\mathbb{D}x)^T + \varepsilon(\mathbb{D}x)P((S-\mathbb{D})x)^T \tag{167}$$

$$+\,\varepsilon((S-\mathbb{D})x)P(\mathbb{D}x)^T + \varepsilon^2((S-\mathbb{D})x)P((S-\mathbb{D})x)\Big]_{ij}\Big)^2 \tag{168}$$

$$\frac{\partial f_3}{\partial \varepsilon}\bigg|_{\varepsilon=0^+} = \frac{1}{\tilde{P}_0}\text{tr}\Big(\bar{S}\Big) - \frac{2}{\tilde{P}_0}\sum_{i=1}^{k}\Bigg(\sum_{j=1}^{k}\Big[(\mathbb{D}x)P(\mathbb{D}x)^T\Big]_{ij} \times \sum_{j=1}^{k}[\bar{S}]_{ij}\Bigg)^2 \tag{169}$$

From here, the proof is followed as in Appendix of Zhao et al. [45] (in Proof of Theorem 3.1) which also borrows from Bickel and Chen [3].

## Appendix G. Complexities of some graph clustering methods

The worst-case time complexity of a loop (i.e. one epoch) in Algorithm 1 is $\mathcal{O}(p^2k + pkn)$ because of the matrix multiplication in the update rule of $C$ (5). Here, $k$ is the number of clusters and $n$ is the feature dimension. Note that $k$ is much smaller than both $p$ and $n$. This makes our method much faster than previous optimization based methods and faster than GCN-based clustering methods which have complexities around $\mathcal{O}(p^2n + pn^2)$.

Some GCN-*based* clustering methods:

- AGC[43] - $\mathcal{O}(p^2nt + ent^2)$
  where $t$ is the number of iterations (within an epoch)
- R-VGAE[30] - $\mathcal{O}(pk^2n + (p(n+k) + e(p+k))$
- S3GC[8] - $\mathcal{O}(pn^2s)$
  where $s$ is the average degree
- HSAN[25] - $\mathcal{O}(pBn)$
  they state it as $\mathcal{O}(B^2d)$ but that is only for 1 batch of size $B$ and not the whole epoch
- VGAECD-OPT[6] - $\mathcal{O}(p^2nD^L)$
  where $D$ is the size of graph filter, $l$ is the number of linear layers

## Appendix H. Integration with GNNs: extended

We iteratively learn the matrix $C$ using gradient descent. From (2.1), we update $X_C$ using the relation $X_C = C^\dagger X$. It is important to note that the loss term $CX_C - X$ will not necessarily be zero. This arises because we use a "soft" version of $C$ ($C_{i,j}$ is the probability $\in [0,1]$) in the loss function to enable gradient flow, while a "hard" version of $C$ ($C_{i,j}$ is the binary assignment $\in \{0,1\}$) is used in the update step. This method ensures that $C$ naturally becomes harder and exhibits a higher prediction probability.

**Q-GCN.** We integrate our loss function ($\mathcal{L}_{MAGC}$, as defined in (3)) into a simple three-layer Graph Convolutional Network (GCN) model. The soft cluster assignments $C$ are learned as the output of the final GCN layer. The detailed architecture and loss function are illustrated in Figure 1.

**Q-VGAE.**

In a VGAE, the encoder learns mean ($\mu$) and variance ($\sigma$): $\mu = \text{GCN}_{\mu}(X, A)$and $\log \sigma = \text{GCN}_{\sigma}(X, A)$ By using the reparameterization trick, we get the distribution of the latent space as: $q(Z|X, A) = \prod_{i=1}^{N} q(\mathbf{z_i}|X, A) = \prod_{i=1}^{N} \mathcal{N}(\mathbf{z_i}|\mu_i, \text{diag}(\sigma_i^2))$ A common choice for decoder is inner-product of the latent space with itself which giving us the reconstructed $\hat{A}$. $p(\hat{A}|Z) = \prod_{i=1}^{p} \prod_{j=1}^{p} p(\hat{A}_{ij}|z_i, z_j)$, with $p(\hat{A}_{ij} = 1|z_i, z_j) = \text{sigmoid}(z_i^T z_j)$ The VGAE loss can be written as $\mathcal{L}_{VGAE} = \lambda_{recon} \underbrace{\mathbb{E}_{q(Z|X,A)}[\log p(\hat{A}|Z)]}_{\text{Reconstruction Error}} - \lambda_{kl} \underbrace{\text{KL}[q(Z|X, A) \,||\, p(Z)]}_{\text{Kullback-Leibler divergence}}$

where, $Z$ represents the latent space of the VGAE, $\hat{A}$ is the reconstructed adjacency matrix, and $\lambda_{recon}$ and $\lambda_{kl}$ are hyperparameters. We add a GCN layer on top of this architecture, which takes $Z$ as input and predicts $C$.

For the VGAE, we minimize the sum of three losses: the reconstruction loss, the KL-divergence loss, and our loss. This combined loss function is expressed as: $\mathcal{L}_{Q-VGAE} = \mathcal{L}_{MAGC} + \mathcal{L}_{VGAE}$

**Q-GMM-VGAE.** This variant of VGAE incorporates a Gaussian Mixture Model (GMM) in the latent space to better capture data distributions. This approach is effective because it minimizes the evidence lower bound (ELBO) or variational lower bound [15, 17, 18] using multiple priors, rather than a single Gaussian prior as in standard VGAE. Hui et al. [15] use a number of priors equal to the number of clusters.

## Appendix I. Dataset Summaries and Metrics

Refer to Table 2 for the dataset summary.

### I.1. Metrics.

A pair of nodes are said to be in agreement if they belong to the same class and are assigned to the same cluster, or they belong to different classes and have been assigned different clusters. For a particular partitioning, ARI is the fraction of agreeable nodes in the graph. Accuracy is obtained by performing a maximum weight bipartite matching between clusters and labels. NMI measures the normalized similarity between the clusters and the labels, and is robust to class imbalances. Mutual Information between two labellings $X$ and $Y$ of the same data is defined as $MI(X, Y) = \sum_{i=1}^{|X|} \sum_{j=1}^{|Y|} \frac{|X_i \cap Y_i|}{N} \log \frac{N|X_i \cap Y_i|}{|X_i||Y_i|}$ and it is scaled between 0 to 1.

| Name | p ($|\mathbf{V}|$) | n ($|X_i|$) | e ($|E|$) | k ($y$) |
|---|---|---|---|---|
| Cora | 2708 | 1433 | 5278 | 7 |
| CiteSeer | 3327 | 3703 | 4614 | 6 |
| PubMed | 19717 | 500 | 44325 | 3 |
| Coauthor CS | 18333 | 6805 | 163788 | 15 |
| Coauthor Physics | 34493 | 8415 | 495924 | 5 |
| Amazon Photo | 7650 | 745 | 238162 | 8 |
| Amazon PC | 13752 | 767 | 491722 | 10 |
| ogbn-arxiv | 169343 | 128 | 1166243 | 40 |
| Brazil | 131 | 0 | 1074 | 4 |
| Europe | 399 | 0 | 5995 | 4 |
| USA | 1190 | 0 | 13599 | 4 |

Table 2: Datasets summary.

### I.2. Training Details

All experiments were run on an NVIDIA A100 GPU and Intel Xeon 2680 CPUs. We are usually running 4-16 experiments together to utilize resources (for example, in 40GB GPU memory, we can run 8 experiments on PubMed simultaneously). Again, the memory costs are more than dominated by the dataset. All experiments used the same environment running CentOS 7, Python 3.9.12, PyTorch 2.0, PyTorch Geometric 2.2.0.

## Appendix J. Results on non-attributed graphs and very large graphs

The results for non-attributed graphs are presented in Table 3 for large datasets are presented in Table 4.

| | Brazil | | | Europe | | | USA | | |
|---|---|---|---|---|---|---|---|---|---|
| **Method** | **ACC $\uparrow$** | **NMI $\uparrow$** | **ARI $\uparrow$** | **ACC $\uparrow$** | **NMI $\uparrow$** | **ARI $\uparrow$** | **ACC $\uparrow$** | **NMI $\uparrow$** | **ARI $\uparrow$** |
| GAE [NeurIPS'16] | 62.6 | 37.8 | 30.8 | 47.6 | 19.9 | 12.7 | 43.9 | 13.6 | 11.8 |
| DGI [ICLR'19] | 64.9 | 31.0 | 30.4 | 48.6 | 16.1 | 12.3 | **52.2** | 22.9 | **21.7** |
| GIC [PAKDD'21] | 40.5 | 23.5 | 14.1 | 40.4 | 9.4 | 6.2 | 49.7 | 22.1 | 19.9 |
| DAEGC [AAAI'19] | 71.0 | **47.4** | 41.2 | 53.6 | 30.9 | 23.3 | 46.4 | **27.2** | 18.4 |
| **Q-GCN (Ours)** | 51.1 | 31.9 | 23.7 | 45.5 | 30.8 | 25.1 | 43.8 | 19.1 | 14.8 |
| VGAE [NeurIPS'16] | 64.1 | 38.0 | 30.7 | 49.9 | 23.5 | 16.7 | 45.8 | 23.1 | 15.7 |
| **Q-VGAE (Ours)** | 50.1 | 35.0 | 19.8 | 46.6 | 19.5 | 17.5 | 46.2 | 19.5 | 16.9 |
| GMM-VGAE [AAAI'20] | 70.2 | 46.0 | 41.9 | 53.1 | 31.1 | 24.4 | 48.1 | 21.9 | 13.2 |
| R-GMM-VGAE [IEEE TKDE'22] | **73.3** | 45.6 | **42.5** | **57.4** | 31.4 | **25.8** | 50.8 | 23.1 | 15.3 |
| **Q-GMM-VGAE (Ours)** | 68.4 | 46.0 | 42.4 | 47.9 | **32.2** | 23.5 | 46.6 | 23.1 | 13.1 |

Table 3: Comparison of all methods on non-attributed datasets using degree.

| | CoauthorCS | | | CoauthorPhysics | | | AmazonPhoto | | | AmazonPC | | | ogbn-arxiv | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Method** | **ACC $\uparrow$** | **NMI $\uparrow$** | **ARI $\uparrow$** | **ACC $\uparrow$** | **NMI $\uparrow$** | **ARI $\uparrow$** | **ACC $\uparrow$** | **NMI $\uparrow$** | **ARI $\uparrow$** | **ACC $\uparrow$** | **NMI $\uparrow$** | **ARI $\uparrow$** | **ACC $\uparrow$** | **NMI $\uparrow$** | **ARI $\uparrow$** |
| FGC | 69.6 | 70.4 | 61.5 | 69.9 | 60.9 | 49.5 | 44.9 | 38.3 | 22.5 | 46.8 | 36.2 | 23.3 | 24.1 | 8.5 | 9.1 |
| **Q-MAGC (Ours)** | 70.2 | 76.4 | 60.2 | 75.3 | 67.2 | 66.1 | 70.4 | 66.6 | **58.6** | **62.4** | 51 | 31.1 | 35.8 | 24.4 | 15.6 |
| **Q-GCN (Ours)** | 85.4 | 79.6 | 79.7 | 85.2 | **72** | **81.6** | 66.3 | 57.6 | 48.3 | 56.7 | 42.4 | 28.8 | 34.4 | 27.1 | 19.7 |
| **Q-VGAE (Ours)** | **85.6** | **79.9** | **81.6** | **86.7** | 69 | 77.7 | 69.0 | 59.4 | 49.0 | 62.3 | 45.7 | **47.2** | **39.5** | 30.4 | **24.7** |
| **Q-GMM-VGAE (Ours)** | 70.1 | 72.5 | 61.6 | 83.1 | 71.5 | 76.9 | **76.8** | **67.1** | 58.3 | 55.5 | **56.4** | 40 | OOM | OOM | OOM |
| DMoN | 68.8 | 69.1 | 57.5 | 45.4 | 56.7 | 50.3 | 61.0 | 63.3 | 55.4 | 45.4 | 49.3 | 47.0 | 25.0 | **35.6** | 12.7 |

Table 4: Comparison of methods on large attributed datasets.

## Appendix K. Ablation Studies

**Comparison of running times** In Figure 2($a$)subfigure, we compare the running times of our method with other baselines. Our method consistently takes less than half the time across all datasets. Notably, on PubMed (large dataset), state-of-the-art methods GMM-VGAE and R-GMM-VGAE (unmodified) require approximately 60 minutes for clustering, whereas our Q-GMM-VGAE delivers superior performance in under 15 minutes, representing a 75% reduction. Additionally, Q-MAGC runs even faster, completing in just 6 minutes on PubMed, and achieves approximately 90% of the performance.

**Modularity Metric Comparison** We treat modularity as a metric and measure the gains observed in modularity over other baselines on the Cora, CiteSeer, and PubMed datasets. We report two types of graph-based metrics: modularity $\mathcal{Q}$ and conductance $\mathcal{C}$, which do not require GT labels. Conductance measures the fraction of total edge volume pointing outside the cluster, with $\mathcal{C}$ being the average conductance across all clusters, where a lower value is preferred.

From Table 2($b$)subfigure, we observe that although DMoN [39] achieves the highest modularity, our method attains significantly higher NMI. For CiteSeer, we achieve a $40\%$ improvement in NMI with only an $8\%$ decrease in modularity, positioning us closer to the ground truth. Additionally, our methods outperform their foundational counterparts, with Q-MAGC outperforming FGC, and Q-VGAE outperforming VGAE.

**Importance of and Evolution of different loss terms**

We analyze the evolution of the different loss terms during training, and also try to measure the impact of each term separately by removing terms from the loss one by one.

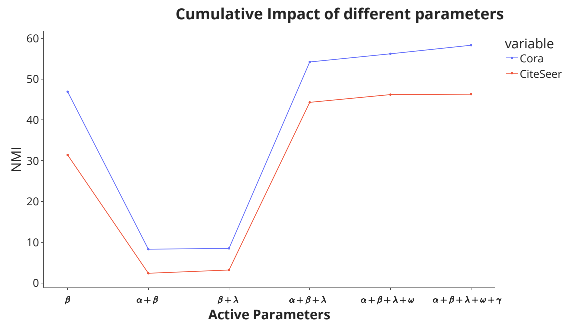
Also, we found that $||CX_C - X||_F^2$ is the most sensitive to change in its weight $\alpha$, followed by the terms related to $\gamma$, $\beta$ and then $\lambda$. This makes sense because if that constraint(relaxation) is not being met, then $C$ would have errors.

Even though some of the terms do the heavy lifting, the other regularization terms do contribute to performance and more importantly, change the nature of $C$ : The smoothness term corresponds to smoothness of signals in the graph being transferred to the coarsened graph which encourages local "patches"/groups of $C$ to belong to the same cluster. The term $\gamma$ ensures that the coarsened graph is connected - i.e. preserving inter-cluster relations, which simple contrastive methods destroy; this affects $C$ by making it so that $\Theta_C$ has minimal multiplicity of 0-eigenvalues.

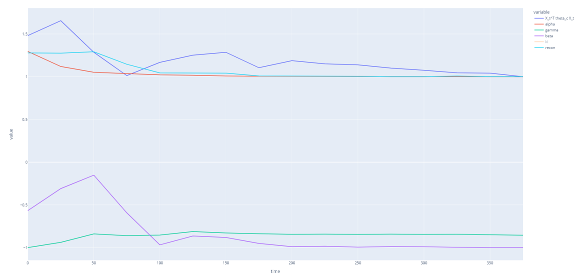
Each separate series has been normalized by its absolute minimum value to see convergence behavior on the same graph easily. Every series is decreasing/converging (except gamma, which represents sparsity regularization and remains almost constant). Thus, we can be assured that no terms are counteracting and hurting the performance. The legend is provided in the graph itself. This plot is on the Cora dataset. 3($b$)subfigure

**Visualization of the latent space**

In Figure 4 and Figure 5, we visualize how the latent space of the Q-VGAE and Q-GMM-VGAE changes over time for various datasets. We use UMAP (Uniform Manifold Approximation and Projection) [29] for dimensionality reduction.

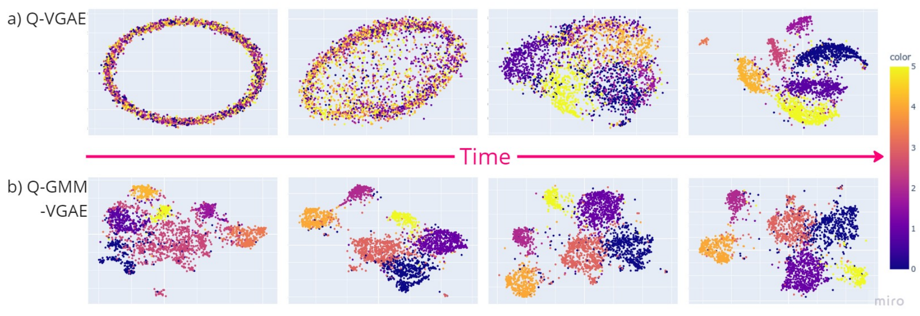


($a$) Impact of active parameters on clustering performance. All terms in the loss (3) are represented by their parameters, for example the modularity term by $\beta$. $\omega$ represents the non-parameterized term.

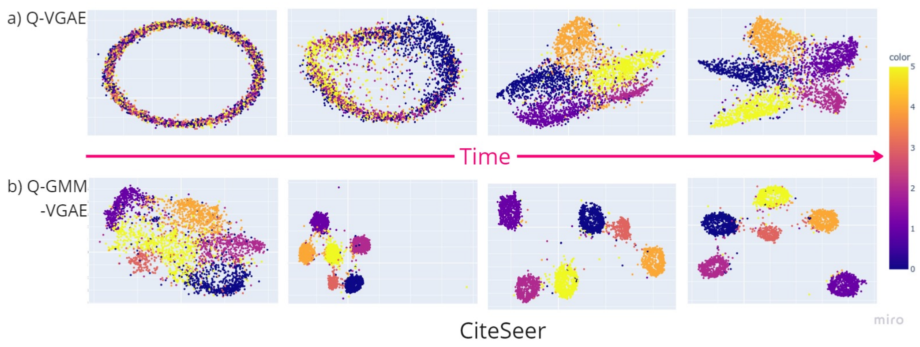


($b$) Evolution of the different loss terms throughout training, denoted by their weight parameters. Also the term `X_t^T theta_C X_t` term is the smoothness term $tr(X_C^T C^T \Theta C X_C)$

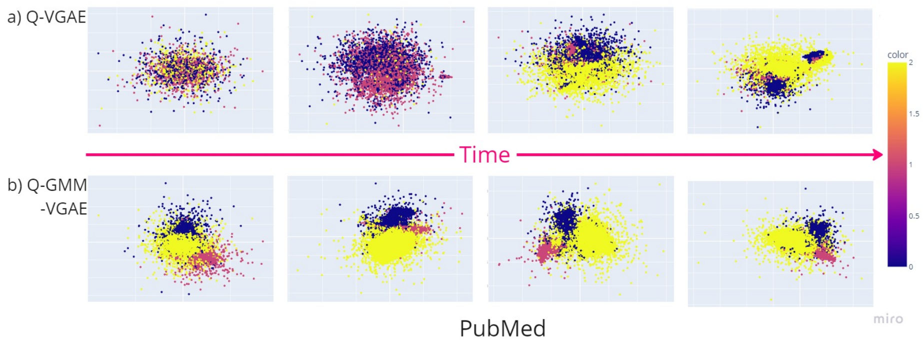
Figure 3



(*a*) Cora



(*b*) CiteSeer



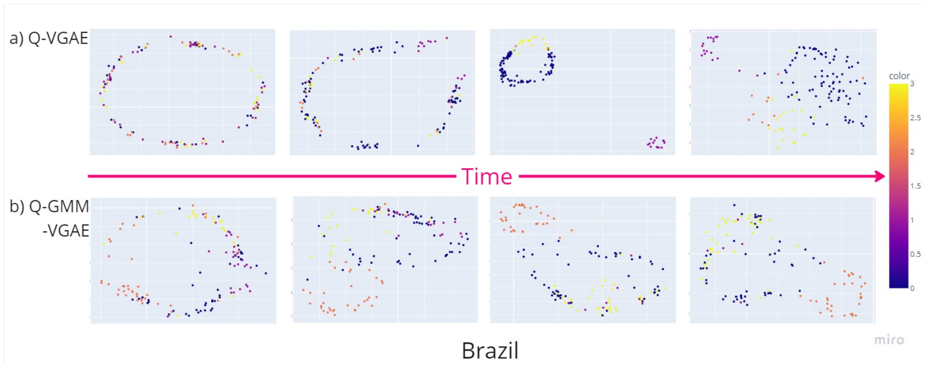
(*c*) PubMed

Figure 4: Plots of evolution of latent space for Q-VGAE and Q-GMM-VGAE methods for Cora, CiteSeer and PubMed datasets. Colors represent clusters.

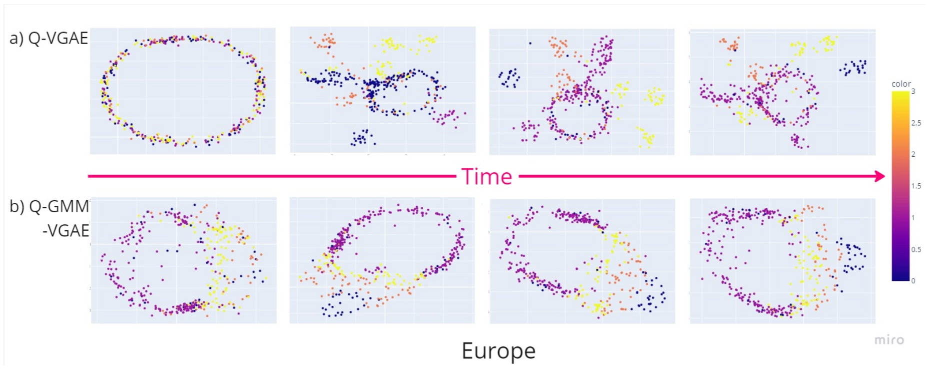
## Appendix L. Attributed SBM theory and results

We validate the robustness and sensitivity of proposed methods to variance in the node features and graph structure. We are also generating features using a multivariate mixture generative model such that the node features of each block are sampled from normal distributions where the centers of clusters are vertices of a hypercube.

**SBM.** The Stochastic Block Model (SBM)[33] is a generative model for graphs that incorporates probabilistic relationships between nodes based on their community assignments. In the basic



($a$) Brazil (Air Traffic)



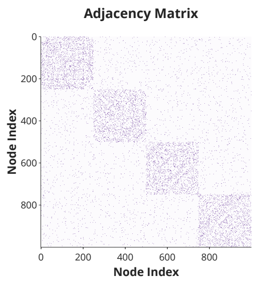
($b$) Europe (Air Traffic)

Figure 5: Plots of evolution of latent space for Q-VGAE and Q-GMM-VGAE methods for Brazil (Air Traffic) and Europe (Air Traffic) datasets.

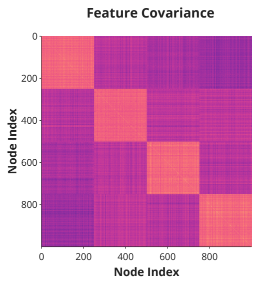
SBM, a network with $p$ nodes is divided into $k$ communities or blocks denoted by $C_i$, where $i = 1, 2, \cdots, k$. The SBM defines a symmetric block probability matrix $B$ with size $(k \times k)$, where each entry $B_{ij}$ represents the probability of an edge between a node in community $C_i$ and a node in community $C_j$. Diagonal entries of this matrix represents the probabilities of intra-cluster edges. This matrix $B$ captures the intra- and inter-community connections and is assumed to be constant. $P(i \leftrightarrow j | C_i = a, C_j = b) = B_{ab}$ denotes the probability of an edge existing between nodes $i$ and $j$ when node $i$ belongs to community $a$ and node $j$ belongs to community $b$. Using these probabilities, the SBM generates a network by independently sampling the presence or absence of an edge for each pair of nodes based on their community assignments and the block probability matrix $B$.

**Degree Corrected SBM.** DC-SBM[16] takes an extra set of parameters $\theta_i$ controlling the expected degree of vertex $i$. Now, the probability of an edge between two nodes (using the same notation as above) becomes $\theta_i \theta_j B_{ab}$. This was introduced to handle the heterogeneity of real-world graphs.

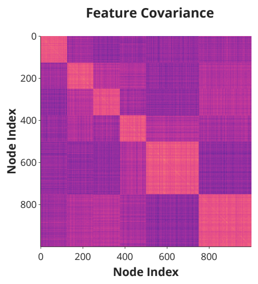
**ADC-SBM Generation.** We make use of the `graph_tool` library to generate the DC-SBM adjacency matrix, with $p = 1000, k = 4$. To generate the $B$ matrix, we follow the procedure in [39], by taking expected degree for each node $d = 20$ and expected sub-degree $d_{out} = 2$. This gives us $B$



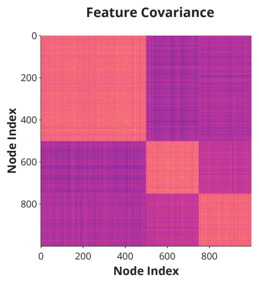
($a$) Adjacency Matrix



($b$) Feature Covariance Matrix $k_f = k$



($c$) Feature Covariance Matrix $k_f > k$



($d$) Feature Covariance Matrix $k_f < k$

Figure 6: Visualization of the generated adjacency and feature covariance matrices for the ADC-SBM

as:

$$\begin{bmatrix} 18 & 2 & 2 & 2 \\ 2 & 18 & 2 & 2 \\ 2 & 2 & 18 & 2 \\ 2 & 2 & 2 & 18 \end{bmatrix}$$

Also, $\theta$ is generated by sampling a power-law distribution with exponent $\alpha = 2$. We constrain the generated vector to $d_{min} = 2$ and $d_{max} = 4$.

To generate features, we use the `make_classification` function in the `sklearn` library. We generate a 128-dimensional feature vector for each node, with no redundant channels. These belong to $k_f$ groups, where $k_f$ might not be equal to $k$. We test three scenarios: a) matched clusters ($k_f = k$) b) nested features ($k_f > k$) c) grouped features ($k_f < k$) as visualized in Figure 6. Note that for better visualization, `class_sep` was increased to 5 (however, the results are given with a value of 1, which is a harder problem).

Additionally, we also consider both cases, with and without the coarsening constraint term.

**Results.** Our objective is able to completely recover the ground truth labels (NMI/ARI/ACC = 1) under all the specified conditions.