The Deep Latent Position Block Model for Block Clustering and Latent Representation of Nodes in Networks

Investigation of Boutin et al. (2025) - Project Report

Karina Musina
karina.musina@student-cs.fr

Yannaël Bossard
yannael.bossard@ens-paris-saclay.fr

Abstract : This document summarizes, contextualizes, and investigates the framework for studying graph-structured data proposed by Boutin et al. [1]. They introduced a novel model, the Deep Latent Position Block Model (Deep LPBM), that simultaneously performs node clustering and interpretable visualization of the network in a latent space. The model is claimed to capture diverse connectivity structures and estimate partial node memberships. In this work, we present a full re-implementation of the Deep LPBM framework from scratch. Furthermore, we propose an extension of the model designed to capture directed graph structures, including self-loops. Our experiments demonstrate that this extension effectively models asymmetric connectivity while maintaining performance comparable to the original architecture on undirected graphs.

1 Introduction

Graph-structured data have become ubiquitous across scientific disciplines. As these networks increase in size and complexity, researchers face the dual challenge of summarization, i.e. grouping nodes to understand connectivity patterns, and visualization, which is essential for ensuring interpretable results. While effectively summarizing and depicting graph topology is a long-standing objective, achieving both simultaneously remains a significant methodological hurdle.

Historically, visualization relied on physical analogies. Eades (1984) modeled graphs as mechanical systems of interacting objects [2]. While these force-directed algorithms produce aesthetic layouts for assortative networks, they inherently struggle to visualize disassortative structures. To address the stochastic nature of connections, Statistical Network Analysis (SNA) emerged, treating edges as outcomes of a probabilistic process rather than deterministic geometric rules.

Within SNA, two primary paradigms have developed, each with distinct limitations. Latent Position Models (LPM) assume nodes exist in a continuous Euclidean space where connection probability decays with distance. While excellent for visualization, standard LPMs are restricted to capturing homophily and cannot easily model complex community structures. Conversely, Block Models, such as the Stochastic Block Model (SBM), assume connectivity is driven strictly by group membership. SBMs excel at detecting diverse patterns, but lack an intrinsic geometric representation, forcing visualization to rely on external, often incompatible, algorithms.

Recent efforts to bridge these paradigms include the Generalized Random Dot Product Graph (GRDPG) and Deep Probabilistic Models. The latter integrates Deep Learning, specifically Graph Neural Networks (GNNs), with probabilistic inference (e.g., Variational Autoencoders) to learn non-linear latent embeddings. However, standard deep approaches like Deep LPM typically employ decoders based on inner products that implicitly enforce transitivity, limiting their ability to detect disassortative patterns.

As a global solution, Boutin et al. (2025) proposes the Deep Latent Position Block Model (Deep

LPBM). This framework introduces a novel block-structured decoder (Deep LPBM) combined with a Graph Convolutional Network (GCN)-based encoder. By assuming edges depend on nodes' partial membership vectors projected into a latent space, Deep LPBM simultaneously provides a network visualization coherent with block modeling and a robust clustering mechanism capable of capturing arbitrary connectivity patterns.

To validate the framework, the authors benchmarked Deep LPBM against state-of-the-art methods across community, hub, and disassortative network topologies. The model matched or outperformed competitors like SBM and Spectral GRDPG, notably succeeding in disassortative settings where standard positional models failed to capture any structural signal . Furthermore, Deep LPBM demonstrated superior capability in estimating partial memberships, allowing it to accurately model nodes with mixed allegiances rather than forcing binary classifications. In a real-world application to the French political blogosphere, the framework produced a highly interpretable visualization that distinctly separated ideological communities while effectively isolating unstructured noise into a specific cluster.

2 Method

The Deep LPBM introduces a variational graph autoencoder (VGAE) framework designed to simultaneously estimate continuous node representations and discrete block structures. The model assumes that the network topology is driven by node partial memberships to latent clusters and a connectivity matrix governing interactions between these clusters.

2.1 Generative Model

Let $G = (\mathcal{V}, \mathbb{E})$ be an undirected graph with N nodes and adjacency matrix A. The model assumes that each node i is associated with a latent vector $z_i \in \mathbb{R}^{Q-1}$, drawn from a standard multivariate normal distribution $p(z_i) = \mathcal{N}(0, I_{Q-1})$.

These latent vectors are mapped to the Q-dimensional simplex Δ_Q via a bijective softmax function to represent *partial memberships* η_i. Specifically, η_{iq} represents the probability that node i belongs to cluster q. Given the partial memberships $\eta = (\eta_i)_{i=1}^N$ and a symmetric connectivity matrix $\Pi \in [0,1]^{Q \times Q}$, edges are sampled independently according to a Bernoulli distribution:

$$P(A_{ij} = 1 | \eta_i, \eta_j, \Pi) = \eta_i^\top \Pi \eta_j = \eta_j^\top \Pi \eta_i \qquad (1)$$

This formulation generalizes the Stochastic Block Model by allowing mixed memberships, enabling the model to capture diverse topological structures such as communities, hubs, and disassortative patterns.

2.2 Variational Graph Autoencoder

Since the posterior distribution $p(Z|A, \Pi)$ is intractable, Deep LPBM employs a variational inference approach.

Encoder: The inference model $q_\phi(Z|A)$ is parameterized by a GCN. The encoder takes the adjacency matrix A (and optionally node features $X = I_N$) as input and outputs the parameters of the variational distribution, assumed to be Gaussian (total factorisation assumption):

$$q_\phi(Z|A) = \prod_{i=1}^{N} \mathcal{N}(z_i; \mu_{\phi,i}(A), \mathrm{diag}(\sigma^2_{\phi,i}(A))) \quad (2)$$

The GCN aggregates neighbor information to compute the means μ_ϕ and variances σ^2_ϕ, effectively embedding structural information into the latent space Z.

Decoder: The decoder reconstructs the graph by estimating the probability of connections using the generative assumption in Eq. 1. The connectivity matrix Π is the primary learnable parameter of the decoder. To ensure constraints, Π is parameterized in unconstrained space via a bijective mapping $f : \mathbb{R} \to (0,1)$ (specifically $f(x) = 0.5 + \pi^{-1} \arctan(x)$) during optimization.

2.3 Optimization and Inference

The model parameters ϕ (encoder weights) and Π (connectivity matrix) are estimated by maximizing the Evidence Lower Bound (ELBO), $\mathcal{L}(\Pi, \phi)$, which approximates the marginal log-likelihood:

$$\begin{aligned} \mathcal{L} &= \mathbb{E}_{q_\phi}[\log p(A|Z,\Pi)] - \mathrm{KL}(q_\phi(Z|A)||p(Z)) \\ &\approx \sum_{i<j} [A_{ij} \log P_{ij} + (1 - A_{ij}) \log(1 - P_{ij})] \\ &\quad - \sum_{i=1}^{N} \mathrm{KL}(\mathcal{N}(\mu_i, \sigma_i^2)||\mathcal{N}(0, I)) \end{aligned} \qquad (3)$$

where $P_{ij} = \eta_i^\top \Pi \eta_j$. The optimization is performed using a gradient-descent algorithm

(Adam). To handle the stochasticity of Z during backpropagation, the *reparameterization trick* is applied: $z_i = \mu_i + \sigma_i \odot \epsilon_i$, where $\epsilon_i \sim \mathcal{N}(0, I)$.

Initialization: To mitigate local optima typical in VAEs, the latent vectors are initialized using a K-Means algorithm on the adjacency matrix. The encoder is then pretrained to match these cluster assignments by minimizing the MSE between the encoder outputs and the K-Means centroids z^0 (obtained via the inverse softmax of the clipped one-hot labels). The pretraining loss is given by:

$$\ell_{\text{init}}(\phi) = \frac{1}{N} \sum_{i=1}^{N} \left(\|\mu_{\phi,i} - z_i^0\|_2^2 + \|\sigma_{\phi,i}^2 - 0.01\|_2^2 \right).$$

Subsequently, the connectivity matrix Π is initialized using the standard SBM approximation based on the initial partial memberships $\eta^0 = \text{softmax}(z^0)$:

$$\Pi_{qr}^0 = \frac{\sum_{i<j} \eta_{iq}^0 \, \eta_{jr}^0 \, A_{ij}}{\sum_{i<j} \eta_{iq}^0 \, \eta_{jr}^0}.$$

2.4 Model Selection

To determine the optimal number of clusters Q, the method employs the Akaike Information Criterion (AIC), defined as:

$$\text{AIC} = \log p(A|\hat{Z}, \hat{\Pi}) - \frac{Q(Q+1)}{2} - N(Q-1) \quad (4)$$

Boutin et al. [1] demonstrate that AIC consistently outperforms BIC and ICL for this specific architecture in recovering the true structural complexity.

3 Implementation

3.1 Reproduction of Results

Due to the unavailability of the official source code, we re-implemented the full Deep LPBM framework from scratch, adhering to the methodology detailed in Boutin et al. [1].

Convergence and Sample Size: We faced challenges reproducing the reported performance using the exact synthetic settings described in the original paper. Specifically, our implementation failed to consistently recover the $Q = 5$ clusters under the setting $N = 200$ and $\beta = 0.3$. The authors claim an estimation of the connectivity matrix $\hat{\Pi}$ with a Frobenius norm difference from Π^* in the order of 10^{-2}. To achieve comparable accuracy with our implementation, we found it necessary to increase the sample size to approximately $N = 350$. These results are illustrated in Fig. 1. We showed in the figure that some instances of the model successfully retrieved $\hat{\Pi}$ with small Frobenius norm difference, leading to a perfect Adjusted Rand Index (ARI) score. We used the estimated $\hat{\eta}$ projected on a 2D space using t-SNE algorithm, leading to interpretable visualization in Fig. 1(d).

Training Dynamics and Stability: We observed that the model is highly sensitive to initialization and the pretraining phase. In most of the trials, the initialization matrix Π^0 exhibited a smaller Frobenius error, and thus a better estimation of connectivity patterns, than the matrix obtained after subsequent training epochs, as illustrated in Fig. 5(d). We show in the table 1 that this effect is consistent across graph-types, as well as across β in Fig. 4. This was obtained with 500 iterations of pre-training and 500 iterations of training. Furthermore, we noted that the ARI performance tended to degrade after prolonged training (Fig. 1(a)). To mitigate this instability, we introduced a learning rate cosine annealing scheduler and carefully tuned the hyperparameters. We also distinguished the encoder from the decoder in the learning rate initial values, as the encoder was already pretrained. Although it led to better stabilization over long training, it did not remove the highly sensitive dependence on initialization and pretraining phase.

Table 1: Comparison of Frobenius norm error on connectivity matrices relative to Π^* for $\beta = 0.4$, $N = 350$.

	Π^0 (Init)	$\hat{\Pi}$ (Estimated)
Assortative	0.028 ± 0.004	0.34 ± 0.02
Disassortative	0.032 ± 0.008	0.09 ± 0.02
Hub	0.033 ± 0.004	0.32 ± 0.03

Visualization: Our experiments suggest that the training phase is primarily beneficial for refining partial memberships rather than visualization. Interestingly, effective visualization could be achieved immediately after pretraining using UMAP projection (Fig. 5(a)). In contrast, t-SNE projection failed to produce meaningful clusters immediately after pretraining (Fig. 5(b)) and required hundreds of additional training iterations to stabilize (Fig. 5(c)).

Model Selection: We compared the Akaike Information Criterion (AIC) and Bayesian Informa-

tion Criterion (BIC) for selecting the number of clusters. Consistent with the findings in the original paper, Fig. 3(a) confirms that AIC correctly identifies the true number of clusters, whereas BIC tends to underperform.

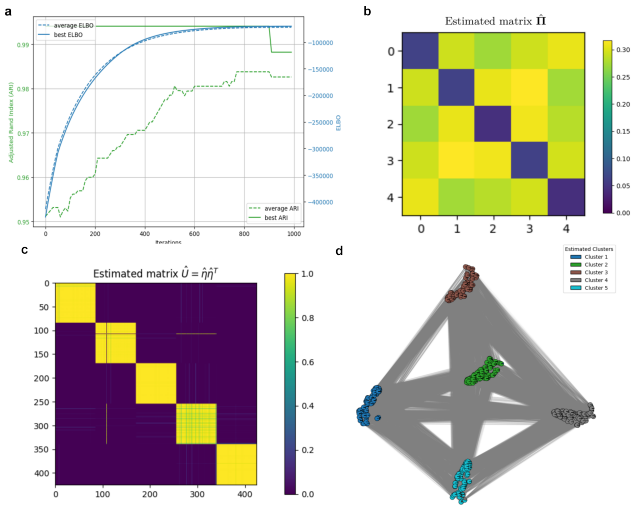

Figure 1: **Deep LPBM results on disassortative adjacency matrix**. **(a) Training Dynamics**: Evolution of ELBO and ARI on a disassortative graph A ($\beta = 0.3, N = 425$). **(b) Estimation of $\hat{\Pi}$**: An instance of the estimated connectivity matrix. **(c) Estimated Partial Memberships**: Matrix $\hat{U}$ derived from $\hat{\eta}$. **(d) Visualization**: t-SNE projection of the estimated latent vectors $\hat{\eta}$.

3.2 Testing on a Real-World Dataset

To evaluate the framework beyond synthetic benchmarks, we applied the Deep LPBM to the SNAP ego-Facebook dataset [3]. This dataset consists of anonymized ego-networks with hand-labeled social circles (e.g., family, colleagues), which serve as ground-truth communities. We anticipate an assortative community structure, where intra-circle connectivity density exceeds inter-circle density.

Experimental Setup: For each ego network, we build the adjacency matrix from the provided undirected edge list and use the annotated circles as ground-truth communities. The circles may overlap and we approximate overlapping labels by assigning uniform fractional memberships across all circles a node belongs to, which may not reflect real social ties.

Results and Limitations: Our experiments highlighted several practical limitations of the Deep LPBM when applied to this empirical data (see Fig. 6):

Unlike in the synthetic regime, the AIC criterion failed to identify a distinct minimum corresponding to the number of social circles (Fig. 6(a-b)). This suggests that the Gaussian latent assumption may not align well with the complex, overlapping topology of real social graphs. Moreover, visual inspection becomes uninformative for dense graphs with several hundred nodes. Additionally, inference is relatively time-consuming, so the method may not be ideal when looking for a good speed-accuracy trade-off; in contrast, a simple initialization such as the first k-means step already provides non-trivial baselines. Because we already observed limitations of the algorithms on synthetic data, e.g. performance degrading with the number of nodes decreasing, it was expected that its application to noisy, real-world networks with overlapping communities struggled more and requires further robustness tuning or stronger regularization.

Finally, the algorithm focuses exclusively on edges. However, in this dataset (and many similar ones), node features are also available and could be incorporated to improve performance, an extension that is not explicitly addressed in the classical Deep LPBM formulation.

3.3 Extension to Directed Graph-Structured Data

The original Deep LPBM framework is explicitly designed for undirected graphs with symmetric connectivity matrices. However, many real-world networks, particularly biological neural networks, are inherently directed. To address this, we extended the framework to support asymmetric connectivity and self-loops.

3.3.1 Methods

Relaxing Symmetry Assumptions: First, we removed the symmetry constraint from the generative model. In the directed setting, the probability of connection becomes asymmetric:

$$P(A_{ij} = 1 | \eta_i, \eta_j, \Pi) = \eta_i^\top \Pi \eta_j \neq \eta_j^\top \Pi \eta_i \qquad (5)$$

Consequently, we modified the decoder's forward pass and initialization to learn an unconstrained matrix Π. Furthermore, the reconstruction loss was adjusted to sum over the entire adjacency matrix A (rather than just the upper triangle), accounting for directed edges and self-connections on the main diagonal.

Graph Attention Encoder: The standard GCN used in the original paper assumes undirected

graphs to ensure the Laplacian matrix is symmetric positive semi-definite [4]. While symmetrizing the input ($A + A^\top$) allows the GCN to run, it discards directionality information, leading to poor performance in our experiments. To address this, we replaced the GCN encoder with a Graph Attention Network (GAT) [5]. GATs employ an attention mechanism that naturally handles asymmetric relationships without requiring valid Laplacian spectral properties.

3.3.2 Results

We evaluated the extended Deep LPBM on synthetic directed graphs designed to mimic biological neural networks. The ground-truth connectivity matrix Π^* consisted of 5 clusters with varying directed connection strengths $\beta \in [0.2, 0.6]$ and noise level $\epsilon = 0.035$ (Fig. 2(a)). Using the ground-truth connectivity, adjacency matrices are generated similarly to the paper, illustrated in Fig. 2(b).

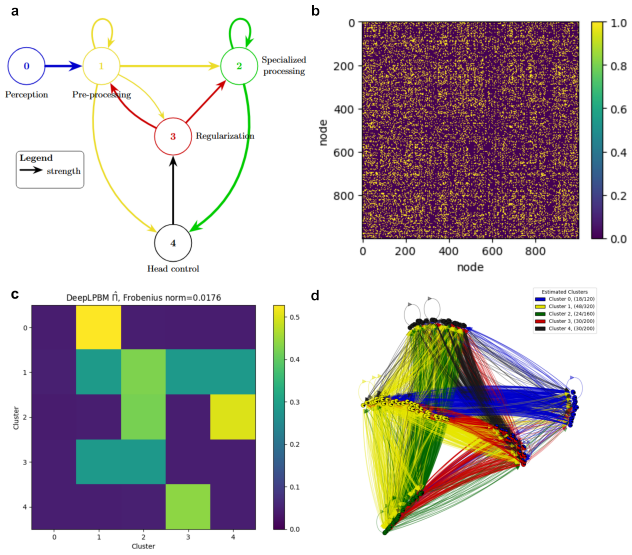

Figure 2: **Deep LPBM extension for directed graphs**. **(a) Ground Truth**: Schematic of the asymmetric connectivity matrix Π^*. **(b) Adjacency Matrix** A: A sample generated from Π^*. **(c) Estimation** $\hat{\Pi}$: The estimated connectivity matrix $\hat{\Pi}$ recovered by our extended model. **(d) Latent Space**: t-SNE visualization of the estimated node embeddings $\hat{\eta}$. As 1000 nodes was non-interpretable, 15% of nodes per cluster were randomly selected for the figure

The extended model successfully recovered the underlying structure. Figure 2(c) shows that the estimated connectivity matrix $\hat{\Pi}$ closely matches the ground truth. Furthermore, the t-SNE projection of the estimated embeddings $\hat{\eta}$ (Fig. 2(d)) demonstrates that the model produces interpretable clusters that respect the directed nature of the data, including the self-loops.

Regarding model selection in this directed setting, while the distinction in the AIC curve (Fig. 3(b)) was less sharp than in the symmetric case, the criterion still correctly identified the optimal number of clusters, confirming the robustness of the metric.

4 Discussion

In this study, we conducted a comprehensive investigation of the Deep LPBM, involving a full reimplementation from scratch and an extension to directed graphs. Our findings validate the model's core premise: the ability to unify block clustering with continuous latent visualization. Specifically, the introduction of a block-structured decoder within a VGAE framework allows for the detection of complex topologies, such as disassortative structures, which traditional positional models often fail to capture.

A significant portion of this work focused on assessing the reproducibility of the results presented by Boutin et al. [1]. Our experiments revealed a substantial gap between the claimed performance and the empirical results obtained under the described settings. We observed high sensitivity to hyperparameters, particularly the initialization of Π and the pretraining phase. Moreover, to achieve the accuracy reported in the paper (e.g., Frobenius norm errors of 10^{-2}), we found it necessary to increase the sample size significantly ($N = 350$ vs. $N = 200$). This suggests that the model's efficiency in low-data regimes may be overstated, or relies on specific tuning strategies not disclosed in the publication.

The absence of official source code presented a major obstacle to verification and implementation. Furthermore, we identified several inconsistencies and typographical errors in the published manuscript (detailed in Sec. 5.2), including conflicting definitions of variational parameters and optimization targets. These discrepancies hinder the exact reproduction of the authors' workflow and suggest that the published results might rely on undocumented heuristics or corrected formulations.

Despite these challenges, our extension to directed graphs demonstrates the flexibility of the underlying framework. By substituting the GCN en-

coder with a GAT and relaxing the symmetry constraints on Π, we successfully adapted Deep LPBM to model asymmetric networks. This extension preserved the model's clustering and visualization capabilities, confirming that the Deep LPBM architecture is robust enough to generalize beyond undirected interactions.

Conclusively, while the Deep LPBM framework offers a compelling unification of block modeling and visualization, its current state requires greater robustness to be considered a "plug-and-play" tool for practitioners. Future work should focus on stabilizing the training dynamics—perhaps via alternative initialization schemes or regularizers—and validating the method on diverse real-world directed datasets. This project underscores the critical importance of code availability and clear documentation in statistical computing to ensure scientific reproducibility.

5 Supplementary Material

5.1 Code Availability

Implementation details, including source code and experimental notebooks, are available in the following GitHub.

5.2 Discrepancies and Clarifications regarding Boutin et al. (2025)

We identified several inconsistencies and typographical errors in the published version of [1]:

- **Dimensionality of W_σ:** On page 7, the authors define the variational parameters as matrices $W_\mu, W_\sigma \in \mathcal{M}_{30\times(Q-1)}(\mathbb{R})$. However, the subsequent text describes the encoder output as *"scalar log standard deviations"*, and the ELBO formulation in Equation (10) implies a scalar variance broadcast across dimensions ($d\sigma^2$). This suggests the correct dimension should be $W_\sigma \in \mathcal{M}_{30\times1}(\mathbb{R})$. In our experiments, we tested both configurations and observed negligible differences in performance.
- **Initialization Loss in Algorithm 1:** A discrepancy exists regarding the initialization strategy for the variational variance. The pseudocode in Algorithm 1 (page 9) minimizes the error on the squared variance: $\|\sigma^2_{\phi,i} - 0.01\|_2^2$. Conversely, the text on page 8 states the minimization is performed between the log-standard deviation $(log\ \sigma_\phi(A)_i)_i$ and the value 0.01. These represent mathematically distinct optimization targets (implying $\sigma \approx 0.1$ versus $\sigma \approx 1.0$, respectively).
- **Formatting of Table 3:** The text on page 13 states that the best results in Table 3 are highlighted in red, blue, and green. However, the table in the published version appears in monochrome. We note that the intended color coding is preserved in the Table 2 of the arXiv preprint version of the article. This is likely a version issue in the published article.

6 Author Contributions

Y.B. wrote the report and contributed to the poster. K.M. prepared the poster and contributed to the report. Y.B. and K.M. developed the code.

References

[1] Rémi Boutin, Pierre Latouche, and Charles Bouveyron. "The Deep Latent Position Block Model for Block Clustering and Latent Representation of Nodes in Networks". In: *Statistics and Computing* 35.5 (2025), p. 151.

[2] Peter Eades. "A heuristic for graph drawing". In: *Congressus numerantium* 42.11 (1984), pp. 149–160.

[3] Julian McAuley and Jure Leskovec. *Learning to Discover Social Circles in Ego Networks.* Vol. 25. 2012, pp. 539–547.

[4] TN Kipf. "Semi-supervised classification with graph convolutional networks". In: *arXiv preprint arXiv:1609.02907* (2016).

[5] Petar Veličković et al. "Graph attention networks". In: *arXiv preprint arXiv:1710.10903* (2017).

A Appendix

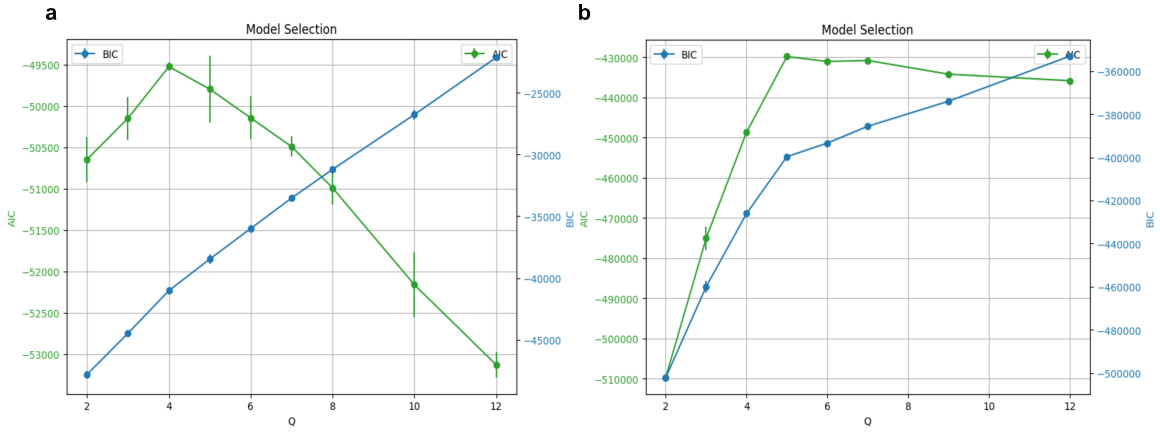

Figure 3: **Selection of the number of clusters**. **(a) Undirected Deep LPBM selection**: AIC and BIC values of Deep LPBM for varying Q on disassortative graph with $Q^* = 5$, N=425 and $\beta = 0.3$. **(b) Directed Deep LPBM selection**:AIC and BIC values of Directed Deep LPBM for varying Q on the generated synthetic biological neural network with $Q^* = 5$, N=1000.

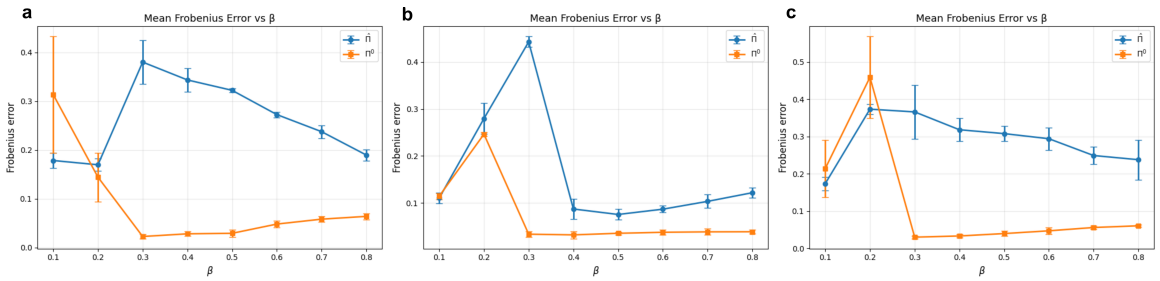

Figure 4: **Comparison of Frobenius norm error** on connectivity matrices relative to Π^* for varying β, $N = 350$. **(a) Assortative graph**. **(b) Disassortative graph**. **(c) Hub graph**.

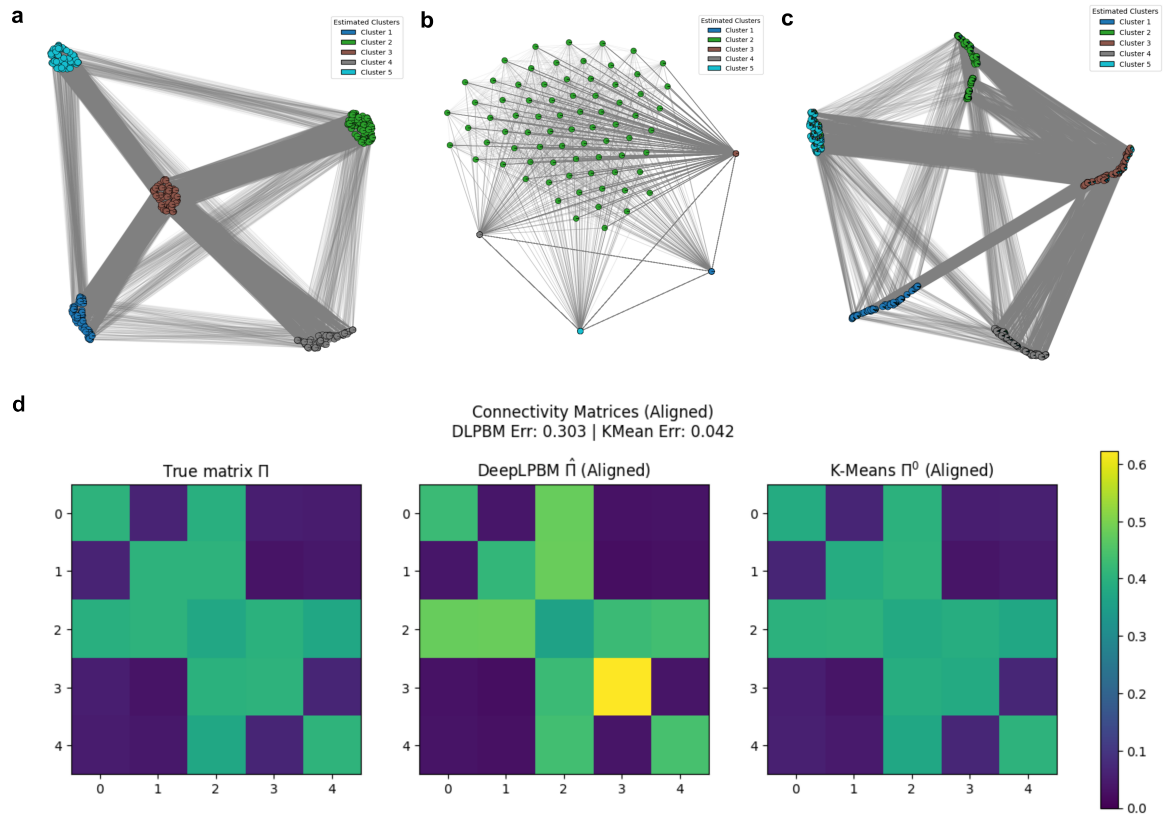

Figure 5: **Deep LPBM instability**. Illustration of the Deep LPBM instability into training phase, on a hub graph A ($\beta = 0.3, N = 425$). **(a) UMAP Latent Space**: Projection of the Deep LPBM partial memberships $\hat{\eta}$ using UMAP projection right after pre-training phase, without training phase. **(b) t-SNE Latent Space**: Projection of the Deep LPBM partial memberships $\hat{\eta}$ using t-SNE projection right after pre-training phase, without training phase. **(c) t-SNE Latent Space**: Projection of the Deep LPBM partial memberships $\hat{\eta}$ using t-SNE projection right after 1000 iterations in training phase. **(d) Estimated of $\hat{\Pi}$**: Illustration of Π^*, $\hat{\Pi}$, and Π^0, with Frobenius norm difference of 0.303 and 0.042, respectively.

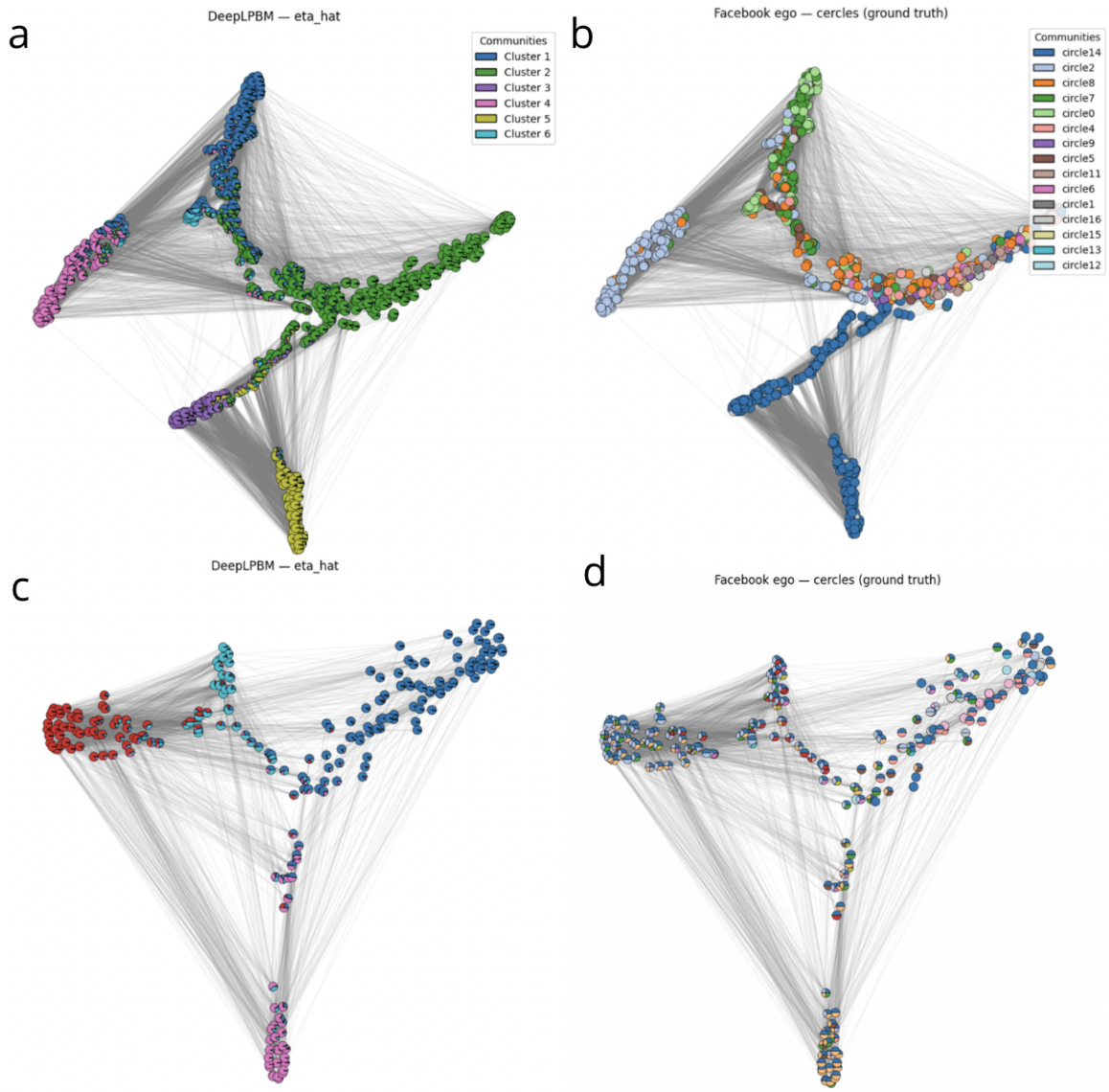

Figure 6: **Tests on egos dataset**. Visualization of the algorithm performance on two different egos. **(a) Ego 1684, Deep LPBM output**: t-SNE projection of the learned partial memberships. **(b) Ego 1684, true circles**: ground-truth circles displayed using the same t-SNE layout. **(c) Ego 348, Deep LPBM output**: same as (a) but for Ego 348. **(d) Ego 348, true circles**: same as (b) but for Ego 348. *Note that Ego 348 exhibits substantial overlap between circles (many nodes belong to several circles), whereas Ego 1684 shows much less intersection between circles.*