Deep Generative Models and Inference Methods for Network Analysis

Seydina Ousmane NIANG

Phd defense in applied mathematics
Friday 2nd October 2026
INRIA Centre Université Côte d'Azur

seydina-ousmane.niang@inria.fr

Supervision: C. Bouveyron, P. Latouche, & M. Corneli



Networks are everywhere

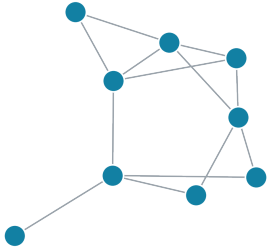
- A network models **relational data**: who interacts with whom.
- Entities become **nodes**.
- Interactions become **edges**.

e.g. a social network: individuals as nodes, relationships as edges.



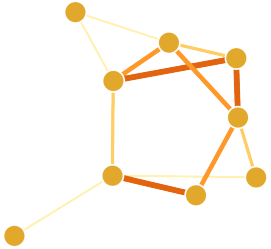
Networks come in several forms

Binary



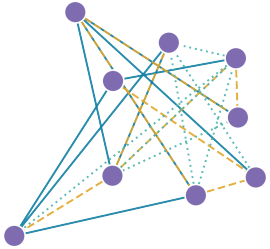
edge present or absent: $A_{ij} \in \{0, 1\}$

Valued



edge carries an intensity: $A_{ij} \in \mathbb{N}$

Multi-view



several interaction types: $A^{(1)}, \ldots, A^{(V)}$

- Real interactions are often richer than binary: weights (valued) or multiple relation types (multi-view).
- Thresholding a valued network or aggregating views **throws away information**.

Analysing a network: three goals, one framework

Given an observed network, we want to model its generation, cluster its nodes, and visualise its structure:



Model

understand *how edges are generated*, probabilistically



Cluster

group nodes by their *connection profiles* into interpretable blocks



Visualise

embed the structure in a *Euclidean space* to make it legible

Analysing a network: three goals, one framework

Given an observed network, we want to model its generation, cluster its nodes, and visualise its structure:



Model

understand *how edges are generated*, probabilistically



Cluster

group nodes by their *connection profiles* into interpretable blocks



Visualise

embed the structure in a *Euclidean space* to make it legible

A generative model $p(A \mid \theta)$ answers all three at once: analysis becomes recovering θ from the observed A.

Analysing a network: three goals, one framework

Given an observed network, we want to model its generation, cluster its nodes, and visualise its structure:

Model

understand *how edges are generated*, probabilistically

Cluster

group nodes by their *connection profiles* into interpretable blocks

Visualise

embed the structure in a *Euclidean space* to make it legible

A generative model $p(A \mid \theta)$ answers all three at once: analysis becomes recovering θ from the observed A.

Advantages of a probabilistic model:

Uncertainty

parameters estimated *with* measures of uncertainty

Model selection

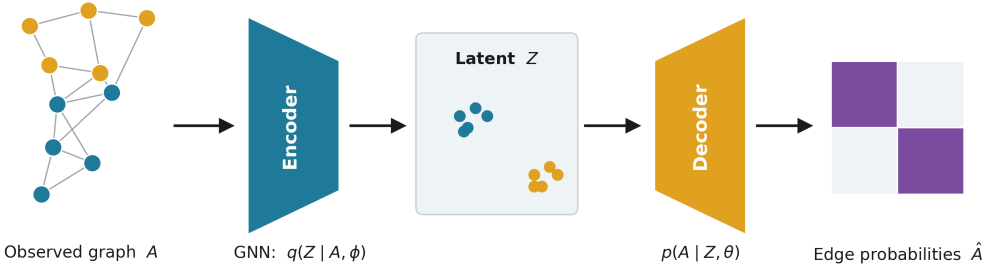
a likelihood lets us *compare* models, e.g. the number of groups

Flexibility

one framework for assortative, disassortative or hub structures

Deep latent variable generative models(Deep-LVM)

(Deep-LVM): an **encoder** embeds the graph, a **decoder** reconstructs it.



The encoder $q(Z \mid A, \phi)$ is modelled as a **graph neural network (GNN)**.

The decoder $p(A \mid Z, \theta)$ is the component that defines the **statistical model** of the network, and is **purpose-built** to the analysis goal.

The **form of the decoder** is the central modelling choice: it determines which structures the model can represent. This thesis revisits it for several type of networks.

Outline

The Deep Latent Position Block Model for the Clustering of Nodes in Multi-graphs

Deep zero-inflated Poisson latent position block model for valued networks

Large-Scale Graphon Learning with Topology-Aware Stochastic Optimization

Conclusion and perspectives

Latent position cluster model (LPCM)

The **LPCM** (Handcock et al., 2007) gives each node a position Z_i in a Euclidean space, and the link probability **decreases with distance**:

$$p(A_{ij} = 1 \mid Z_i, Z_j) = \sigma\big(\alpha - \|Z_i - Z_j\|\big).$$

- **Decoder:** distance-based, so *close nodes connect* with high probability.
- **Clustering:** the positions Z_i are drawn from a Gaussian mixture, each component being a cluster:

  $$Z_i \sim \sum_{q=1}^{Q} \lambda_q \mathcal{N}(\mu_q, \Sigma_q),$$

  so Z carries **both the visualisation and the clustering**.
- **Limitation:** only assortative structure. It **cannot represent heterogeneous patterns**.

Stochastic block model (SBM)

The **SBM** (Holland et al., 1983) gives each node a hard block label $C_i \in \{1, \ldots, Q\}$, and the link probability depends only on the two blocks:

$$p(A_{ij} = 1 \mid C_i = q, C_j = r) = \pi_{qr}.$$

- **Decoder:** a block to block connectivity matrix π.
- **Strength:** the connectivity matrix π is unconstrained, so the SBM captures *any* block-to-block pattern: assortative, disassortative, hubs.
- **Limitation:** the latent space is discrete (a hard label per node), so it offers no Euclidean embedding to visualise the network.

Deep-latent position block model (LPBM)

Deep-LPBM (Boutin et al., 2025) combines both: a continuous latent position *and* a block matrix. Positions map to partial memberships $\gamma_i \in \Delta_Q$ (bijective softmax), and links use a block matrix over memberships:

$$p(A_{ij} = 1 \mid Z_i, Z_j) = \gamma_i^\top \pi \gamma_j.$$

- **Best of both:** a Euclidean embedding (from the positions) *and* heterogeneous structure (from the block matrix π), plus overlapping clusters via partial memberships.
- **Limitation:** only binary networks.

Deep-latent position block model (LPBM)

Deep-LPBM (Boutin et al., 2025) combines both: a continuous latent position *and* a block matrix. Positions map to partial memberships $\gamma_i \in \Delta_Q$ (bijective softmax), and links use a block matrix over memberships:

$$p(A_{ij} = 1 \mid Z_i, Z_j) = \gamma_i^\top \pi \gamma_j.$$

- **Best of both:** a Euclidean embedding (from the positions) *and* heterogeneous structure (from the block matrix π), plus overlapping clusters via partial memberships.
- **Limitation:** only binary networks.

Deep-LPBMM extends the LPBM to multi-view networks.

Generative model: assumptions about the generation of networks

- We observe L views on a common set of size N. Each view $\ell \in \{1, \dots, L\}$ is modelled by a symmetric adjacency matrix A^ℓ, with

$$A_{ij}^\ell = \begin{cases} 1, & \text{if nodes } i \text{ and } j \text{ are connected in view } \ell, \\ 0, & \text{otherwise.} \end{cases}$$

Generative model: assumptions about the generation of networks

- We observe L views on a common set of size N. Each view $\ell \in \{1, \dots, L\}$ is modelled by a symmetric adjacency matrix A^ℓ, with

$$A^\ell_{ij} = \begin{cases} 1, & \text{if nodes } i \text{ and } j \text{ are connected in view } \ell, \\ 0, & \text{otherwise.} \end{cases}$$

- All views in $A := \{A^{(1)}, \dots, A^{(L)}\}$ are explained by a single latent representation, drawn for each node i as

$$Z_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}_d(0, I_d), \qquad d = Q - 1.$$

Generative model: assumptions about the generation of networks

- We observe L views on a common set of size N. Each view $\ell \in \{1, \dots, L\}$ is modelled by a symmetric adjacency matrix A^ℓ, with

$$A_{ij}^\ell = \begin{cases} 1, & \text{if nodes } i \text{ and } j \text{ are connected in view } \ell, \\ 0, & \text{otherwise.} \end{cases}$$

- All views in $A := \{A^{(1)}, \dots, A^{(L)}\}$ are explained by a single latent representation, drawn for each node i as

$$Z_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}_d(0, I_d), \qquad d = Q - 1.$$

- Nodes belong to the Q clusters in a non-exclusive manner: a bijective softmax $h : \mathbb{R}^{Q-1} \to \Delta_Q$ maps the latent positions onto **partial memberships** probabilities

$$\gamma_{iq} := \big[h(Z_i)\big]_q = \begin{cases} \dfrac{\exp(Z_{iq})}{1 + \sum_{r=1}^{Q-1} \exp(Z_{ir})}, & q \neq Q, \\ \dfrac{1}{1 + \sum_{r=1}^{Q-1} \exp(Z_{ir})}, & q = Q, \end{cases}$$

so that $\gamma_{iq} = \Pr(i \in \text{cluster } q)$ and $\gamma_i \in \Delta_Q$.

Generative model: the decoder

- Connection between nodes i and j in view ℓ:

$$A_{ij}^{(\ell)} \mid \gamma_i, \gamma_j \overset{\text{i.i.d.}}{\sim} \mathcal{B}\left(\gamma_i^\top \pi^{(\ell)} \gamma_j\right),$$

where $\pi^{(\ell)}$ is the symmetric $Q \times Q$ connectivity matrix of view ℓ. $\pi := \{\pi^{(\ell)}\}_\ell$,
$\gamma_i^\top \pi^{(\ell)} \gamma_j = \sum_{q,r} \gamma_{iq}^\top \pi_{qr}^{(\ell)} \gamma_{jr}$.

Generative model: the decoder

- Connection between nodes i and j in view ℓ:

$$A_{ij}^{(\ell)} \mid \gamma_i, \gamma_j \overset{\text{i.i.d.}}{\sim} \mathcal{B}\left(\gamma_i^\top \pi^{(\ell)} \gamma_j\right),$$

  where $\pi^{(\ell)}$ is the symmetric $Q \times Q$ connectivity matrix of view ℓ. $\pi := \{\pi^{(\ell)}\}_\ell$,
  $\gamma_i^\top \pi^{(\ell)} \gamma_j = \sum_{q,r} \gamma_{iq}^\top \pi_{qr}^{(\ell)} \gamma_{jr}$.

- Views are independent given Z:

$$p(\mathbf{A}, \mathbf{Z} \mid \pi) = \prod_{\ell=1}^{L} p(\mathbf{A}^{(\ell)} \mid \mathbf{Z}, \pi^{(\ell)})\, p(\mathbf{Z}).$$

Generative model: the decoder

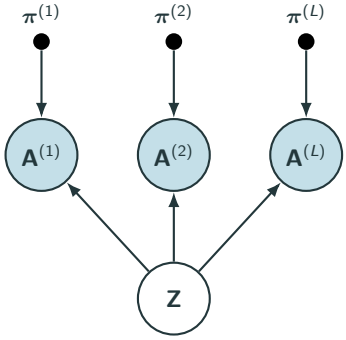
- Connection between nodes i and j in view ℓ:

$$A_{ij}^{(\ell)} \mid \gamma_i, \gamma_j \overset{\text{i.i.d.}}{\sim} \mathcal{B}\left(\gamma_i^\top \pi^{(\ell)} \gamma_j\right),$$

where $\pi^{(\ell)}$ is the symmetric $Q \times Q$ connectivity matrix of view ℓ. $\pi := \{\pi^{(\ell)}\}_\ell$,
$\gamma_i^\top \pi^{(\ell)} \gamma_j = \sum_{q,r} \gamma_{iq}^\top \pi_{qr}^{(\ell)} \gamma_{jr}$.

- Views are independent given Z:

$$p(\mathbf{A}, \mathbf{Z} \mid \pi) = \prod_{\ell=1}^{L} p(\mathbf{A}^{(\ell)} \mid \mathbf{Z}, \pi^{(\ell)})\, p(\mathbf{Z}).$$



Graphical model of Deep-LPBMM.

Inference: from an intractable likelihood to the ELBO

Goal: maximise $\log p(A \mid \pi)$ $\quad (\rightarrow$ intractable$)$.

Variational inference, for any $q(Z)$:

$$\log p(A \mid \pi) = \underbrace{\mathbb{E}_{q(Z)}\big[\log p(A, Z \mid \pi)\big] - \mathbb{E}_{q(Z)}\big[\log q(Z)\big]}_{\mathcal{L}(q,\pi)\ \textbf{(ELBO)}} + \underbrace{\mathrm{KL}\big(q(Z) \,\|\, p(Z \mid A, \pi)\big)}_{\geq 0}.$$

maximise ELBO $\Leftrightarrow$ q close to the posterior

Two terms:

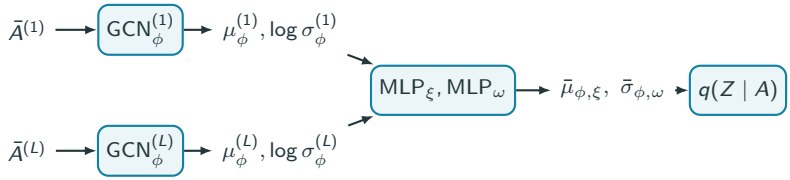
$$\mathcal{L}(q,\pi) = \underbrace{\sum_{\ell=1}^{L} \mathbb{E}_q\big[\log p(A^{(\ell)} \mid Z, \pi^{(\ell)})\big]}_{\text{reconstruction}} - \underbrace{\mathrm{KL}\big(q(Z \mid A) \,\|\, p(Z)\big)}_{\text{regularisation}}.$$

Inference: the encoder

We restrict q to a mean-field Gaussian whose parameters are predicted by a neural network (*amortised* inference):

$$q(Z \mid A) = \prod_{i=1}^{N} \mathcal{N}\big(Z_i;\ \bar{\mu}_{\phi,\xi}(A)_i,\ \bar{\sigma}_{\phi,\omega}(A)_i^2\, I_d\big).$$

Per view, a **graph convolutional network** (GCN, Kipf and Welling, 2017) reads the normalised $\bar{A}^{(\ell)}$ and outputs $(\mu_\phi^{(\ell)}, \log \sigma_\phi^{(\ell)})$; two **MLPs** merge the views into a single shared representation.



Inference: optimisation by gradient ascent

$$\mathcal{L}(q, \pi) = \sum_{\ell=1}^{L} \mathbb{E}_q\big[\, \log p(A^{(\ell)} \mid Z, \pi^{(\ell)})\big] \;-\; \mathrm{KL}\big(q(Z \mid A) \,\|\, p(Z)\big).$$

Inference: optimisation by gradient ascent

$$\mathcal{L}(q, \pi) = \sum_{\ell=1}^{L} \mathbb{E}_q\big[\log p(A^{(\ell)} \mid Z, \pi^{(\ell)})\big] \; - \; \mathrm{KL}\big(q(Z \mid A) \,\|\, p(Z)\big).$$

We maximise the ELBO $\mathcal{L}(q, \pi)$ by gradient ascent w.r.t. (π, ϕ, ξ, ω).

Inference: optimisation by gradient ascent

$$\mathcal{L}(q, \pi) = \sum_{\ell=1}^{L} \mathbb{E}_q\big[\log p(A^{(\ell)} \mid Z, \pi^{(\ell)})\big] \; - \; \mathrm{KL}\big(q(Z \mid A) \,\|\, p(Z)\big).$$

We maximise the ELBO $\mathcal{L}(q, \pi)$ by gradient ascent w.r.t. (π, ϕ, ξ, ω).

- In π: **easy**, since q does not depend on it, the gradient moves inside the expectation and is estimated by sampling.

Inference: optimisation by gradient ascent

$$\mathcal{L}(q, \pi) = \sum_{\ell=1}^{L} \mathbb{E}_q\big[\log p(A^{(\ell)} \mid Z, \pi^{(\ell)})\big] \; - \; \mathrm{KL}\big(q(Z \mid A) \,\|\, p(Z)\big).$$

We maximise the ELBO $\mathcal{L}(q, \pi)$ by gradient ascent w.r.t. (π, ϕ, ξ, ω).

- In π: **easy**, since q does not depend on it, the gradient moves inside the expectation and is estimated by sampling.
- In (ϕ, ξ, ω): delicate. The reconstruction expectation is over q_ϕ, whose *distribution* depends on the parameters we differentiate, so $\nabla_\phi \mathbb{E}_{q_\phi}[f(Z)] \neq \mathbb{E}_{q_\phi}[\nabla_\phi f(Z)]$.

Inference: optimisation by gradient ascent

$$\mathcal{L}(q,\pi) = \sum_{\ell=1}^{L} \mathbb{E}_q\big[\log p(A^{(\ell)} \mid Z, \pi^{(\ell)})\big] \;-\; \mathrm{KL}\big(q(Z \mid A) \,\|\, p(Z)\big).$$

We maximise the ELBO $\mathcal{L}(q,\pi)$ by gradient ascent w.r.t. (π, ϕ, ξ, ω).

- In π: **easy**, since q does not depend on it, the gradient moves inside the expectation and is estimated by sampling.
- In (ϕ, ξ, ω): delicate. The reconstruction expectation is over q_ϕ, whose *distribution* depends on the parameters we differentiate, so $\nabla_\phi \mathbb{E}_{q_\phi}[f(Z)] \neq \mathbb{E}_{q_\phi}[\nabla_\phi f(Z)]$.

Reparameterisation trick. Move the randomness into a *parameter-free* noise,

$$Z_i^{(s)} = \bar{\mu}_{\phi,\xi}(A)_i + \bar{\sigma}_{\phi,\omega}(A)_i \odot \epsilon^{(s)}, \qquad \epsilon^{(s)} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, I_{Q-1}),$$

so Z becomes a **differentiable** function of the parameters and the gradient passes inside the expectation: $\nabla_\phi \, \mathbb{E}_{q_\phi}[f(Z)] = \mathbb{E}_\epsilon\big[\nabla_\phi f(\bar{\mu} + \bar{\sigma} \odot \epsilon)\big]$.

Synthetic experiments: network structures

Each view generated by SBM: the link probability depends only on the clusters, through the connectivity matrix π ($\beta > \delta$; the larger β, the easier the recovery).

Community

high within, low between

$$\pi = \begin{pmatrix} \beta & \delta & \cdots & \delta \\ \delta & \beta & \cdots & \delta \\ \vdots & \vdots & \ddots & \vdots \\ \delta & \delta & \cdots & \beta \end{pmatrix}$$

$$(\beta - \delta) I_Q + \delta \mathbf{1}_Q$$

Disassortative

low within, high between

$$\pi = \begin{pmatrix} \delta & \beta & \cdots & \beta \\ \beta & \delta & \cdots & \beta \\ \vdots & \vdots & \ddots & \vdots \\ \beta & \beta & \cdots & \delta \end{pmatrix}$$

$$(\delta - \beta) I_Q + \beta \mathbf{1}_Q$$

Hub

one cluster links to all

$$\pi = \begin{pmatrix} \beta & \beta & \cdots & \beta \\ \beta & \beta & \cdots & \delta \\ \vdots & \vdots & \ddots & \vdots \\ \beta & \delta & \cdots & \beta \end{pmatrix}$$

Evaluation. Each node is assigned to the cluster with the highest partial membership; accuracy is measured by the adjusted rand index (ARI).

Synthetic experiments: benchmark (ARI)

Compared with multi-view competitors (MSBM, mimi-SBM) and single-view models (SBM, Deep-LPBM). All methods are given the true number of clusters. Networks size is $N = 200$.

Scenario 1 Complementary

	$\beta = 0.2$	$\beta = 0.3$
Deep-LPBM	0.35	0.39
SBM uni-view	0.40	0.42
MSBM multi-view	1.00	1.00
mimi-SBM	1.00	1.00
Deep-LPBMM	**1.00**	**1.00**

Scenario 2 redundant

	$\beta = 0.1$	$\beta = 0.2$	$\beta = 0.3$
Deep-LPBM	0.14	0.64	0.87
SBM uni-view	0.19	0.78	0.79
MSBM multi-view	0.34	1.00	1.00
mimi-SBM	0.27	0.58	0.78
Deep-LPBMM	**0.35**	**1.00**	**1.00**

Scenario 3 different structure ($\beta = 0.2$)

	Communities	Disassortative	Hub
Deep-LPBM	0.72	0.29	0.71
SBM uni-view	0.80	0.79	0.80
MSBM multi-view	1.00	1.00	1.00
mimi-SBM	1.00	1.00	1.00
Deep-LPBMM	**1.00**	**0.99**	**1.00**



The Deep Latent Position Block Model for the Clustering of Nodes in Multi-graphs.
Niang, Bouveyron, Corneli, Latouche, Boutin. **Journal of Classification**, 2024.

Outline

The Deep Latent Position Block Model for the Clustering of Nodes in Multi-graphs

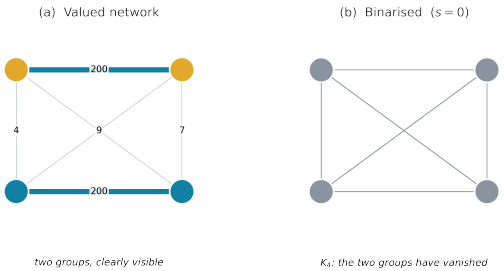
Deep zero-inflated Poisson latent position block model for valued networks

Large-Scale Graphon Learning with Topology-Aware Stochastic Optimization

Conclusion and perspectives

Why valued networks matter

- Edges carry an intensity: strength, frequency, capacity ($Y_{ij} \in \mathbb{N}$ for counts).
- Binarising with a threshold s **discards the magnitude**, often the most informative part of the signal.
- The threshold is **arbitrary**: the recovered structure depends on it.



A two-group network, erased by binarisation.

Generative model: the zero-inflated decoder

A valued, directed network is a non-symmetric adjacency matrix $A \in \mathbb{N}^{N \times N}$, with A_{ij} the number of interactions from i to j.

Generative model: the zero-inflated decoder

A valued, directed network is a non-symmetric adjacency matrix $A \in \mathbb{N}^{N \times N}$, with A_{ij} the number of interactions from i to j.

- Each node i has a latent position $Z_i \sim \mathcal{N}(0, \nu^2 I_{Q-1})$.

Generative model: the zero-inflated decoder

A valued, directed network is a non-symmetric adjacency matrix $A \in \mathbb{N}^{N \times N}$, with A_{ij} the number of interactions from i to j.

- Each node i has a latent position $Z_i \sim \mathcal{N}(0, \nu^2 I_{Q-1})$.
- It belongs to the Q clusters non-exclusively, with partial memberships $\gamma_i = \mathrm{Softmax}(Z_i) \in \Delta_Q$.

Generative model: the zero-inflated decoder

A valued, directed network is a non-symmetric adjacency matrix $A \in \mathbb{N}^{N \times N}$, with A_{ij} the number of interactions from i to j.

- Each node i has a latent position $Z_i \sim \mathcal{N}(0, \nu^2 I_{Q-1})$.
- It belongs to the Q clusters non-exclusively, with partial memberships $\gamma_i = \mathrm{Softmax}(Z_i) \in \Delta_Q$.
- Connections depend only on cluster memberships. Given Z, nodes i and j are linked with probability $\gamma_i^\top \Pi \gamma_j$.

Generative model: the zero-inflated decoder

A valued, directed network is a non-symmetric adjacency matrix $A \in \mathbb{N}^{N \times N}$, with A_{ij} the number of interactions from i to j.

- Each node i has a latent position $Z_i \sim \mathcal{N}(0, \nu^2 I_{Q-1})$.
- It belongs to the Q clusters non-exclusively, with partial memberships $\gamma_i = \mathrm{Softmax}(Z_i) \in \Delta_Q$.
- Connections depend only on cluster memberships. Given Z, nodes i and j are linked with probability $\gamma_i^\top \Pi \gamma_j$.
- Given that a link exists, its weight is $\mathcal{P}(\gamma_i^\top \Lambda \gamma_j)$.

Generative model: the zero-inflated decoder

A valued, directed network is a non-symmetric adjacency matrix $A \in \mathbb{N}^{N \times N}$, with A_{ij} the number of interactions from i to j.

- Each node i has a latent position $Z_i \sim \mathcal{N}(0, \nu^2 I_{Q-1})$.
- It belongs to the Q clusters non-exclusively, with partial memberships $\gamma_i = \mathrm{Softmax}(Z_i) \in \Delta_Q$.
- Connections depend only on cluster memberships. Given Z, nodes i and j are linked with probability $\gamma_i^\top \Pi \gamma_j$.
- Given that a link exists, its weight is $\mathcal{P}(\gamma_i^\top \Lambda \gamma_j)$.

This defines a zero-inflated Poisson decoder:

$$A_{ij} \mid Z \sim \mathrm{ZIP}\big(\underbrace{\gamma_i^\top \Pi \gamma_j}_{\text{existence}}, \underbrace{\gamma_i^\top \Lambda \gamma_j}_{\text{intensity}}\big), \quad i \neq j,$$

$$\begin{cases} \mathbb{P}(A_{ij} = 0 \mid Z) = (1 - \gamma_i^\top \Pi \gamma_j) + \gamma_i^\top \Pi \gamma_j \, \exp(-\gamma_i^\top \Lambda \gamma_j), \\ \mathbb{P}(A_{ij} = k \mid Z) = \gamma_i^\top \Pi \gamma_j \, \dfrac{(\gamma_i^\top \Lambda \gamma_j)^k}{k!} \exp(-\gamma_i^\top \Lambda \gamma_j), \quad k \geq 1. \end{cases}$$

Inference: a directed encoder

The variational family is the same amortised mean-field Gaussian as in Deep-LPBMM. The first contribution here is the **encoder**: a directed graph has in- and out-neighbourhoods that must be read *separately*, not symmetrised away.

- Split A into its **upper** and **lower** triangular parts, A_{up} and A_{low}: they carry the two directions of interaction.
- Same preprocessing as in standard GCNs:

$$\bar{A}_{\rightarrow} = \tilde{D}_{\rightarrow}^{-\frac{1}{2}}(A_{\mathrm{up}} + A_{\mathrm{up}}^{\top} + I_N)\tilde{D}_{\rightarrow}^{-\frac{1}{2}}, \qquad \bar{A}_{\leftarrow} = \tilde{D}_{\leftarrow}^{-\frac{1}{2}}(A_{\mathrm{low}} + A_{\mathrm{low}}^{\top} + I_N)\tilde{D}_{\leftarrow}^{-\frac{1}{2}}.$$

- The encoder input is the concatenation $\bar{A} = [\bar{A}_{\rightarrow}, \bar{A}_{\leftarrow}] \in \mathbb{R}^{N \times 2N}$: each node is described by both its out- and in-connectivity.

Inference: a directed encoder

We designed a GCN-inspired architecture f_ϕ that reads the concatenated directional adjacency $\bar{A} = [\bar{A}_{\rightarrow}, \bar{A}_{\leftarrow}]$ and outputs, for each node, the **mean** and **variance** of its latent position:

$$[\mu_\phi(\bar{A}),\ \log \sigma_\phi(\bar{A})] = f_\phi(\bar{A}).$$

Unlike a standard GCN, it aggregates in- and out-neighbours separately, so **directionality is preserved**.

This defines the amortised posterior

$$q(Z \mid A) = \prod_{i=1}^{N} \mathcal{N}\big(Z_i;\ \mu_\phi(\bar{A})_i,\ \sigma^2_\phi(\bar{A})_i\, \mathbb{I}_{Q-1}\big).$$

Inference: tightening the bound with the iw-ELBO

The latent variable lives on the simplex (Z constrained to a $Q-1$ manifold), so the mean-field posterior is a rough fit and the standard ELBO is a **loose** surrogate for $\log p(A \mid \Pi, \Lambda)$.

Inference: tightening the bound with the iw-ELBO

The latent variable lives on the simplex (Z constrained to a $Q-1$ manifold), so the mean-field posterior is a rough fit and the standard ELBO is a **loose** surrogate for $\log p(A \mid \Pi, \Lambda)$.

Starting from the exact marginal likelihood,

$$\log p(A \mid \Pi, \Lambda) = \log \mathbb{E}_{Z \sim q}\left[\frac{p(A, Z \mid \Pi, \Lambda)}{q(Z \mid A)}\right],$$

Inference: tightening the bound with the iw-ELBO

The latent variable lives on the simplex (Z constrained to a $Q-1$ manifold), so the mean-field posterior is a rough fit and the standard ELBO is a **loose** surrogate for $\log p(A \mid \Pi, \Lambda)$.

Starting from the exact marginal likelihood,

$$\log p(A \mid \Pi, \Lambda) = \log \mathbb{E}_{Z \sim q}\left[\frac{p(A, Z \mid \Pi, \Lambda)}{q(Z \mid A)}\right],$$

the importance-weighted ELBO (Burda et al., 2016) averages K samples *inside* the log:

$$\mathcal{L}_K = \mathbb{E}_{Z^1, \ldots, Z^K \overset{\text{i.i.d.}}{\sim} q}\left[\log \frac{1}{K} \sum_{\ell=1}^{K} \frac{p(A, Z^\ell \mid \Pi, \Lambda)}{q(Z^\ell \mid A)}\right].$$

Inference: tightening the bound with the iw-ELBO

The latent variable lives on the simplex (Z constrained to a $Q-1$ manifold), so the mean-field posterior is a rough fit and the standard ELBO is a **loose** surrogate for $\log p(A \mid \Pi, \Lambda)$.

Starting from the exact marginal likelihood,

$$\log p(A \mid \Pi, \Lambda) = \log \mathbb{E}_{Z \sim q}\left[\frac{p(A, Z \mid \Pi, \Lambda)}{q(Z \mid A)}\right],$$

the importance-weighted ELBO (Burda et al., 2016) averages K samples *inside* the log:

$$\mathcal{L}_K = \mathbb{E}_{Z^1, \ldots, Z^K \overset{\text{i.i.d.}}{\sim} q}\left[\log \frac{1}{K} \sum_{\ell=1}^{K} \frac{p(A, Z^\ell \mid \Pi, \Lambda)}{q(Z^\ell \mid A)}\right].$$

- **Tighter bound**: ELBO $= \mathcal{L}_1 \leq \mathcal{L}_K \leq \mathcal{L}_{K+1} \leq \log p(A \mid \Pi, \Lambda)$, converging to $\log p(A)$ as $K \to \infty$.

Inference: tightening the bound with the iw-ELBO

The latent variable lives on the simplex (Z constrained to a $Q-1$ manifold), so the mean-field posterior is a rough fit and the standard ELBO is a **loose** surrogate for $\log p(A \mid \Pi, \Lambda)$.

Starting from the exact marginal likelihood,

$$\log p(A \mid \Pi, \Lambda) = \log \mathbb{E}_{Z \sim q}\left[\frac{p(A, Z \mid \Pi, \Lambda)}{q(Z \mid A)}\right],$$

the importance-weighted ELBO (Burda et al., 2016) averages K samples *inside* the log:

$$\mathcal{L}_K = \mathbb{E}_{Z^1, \ldots, Z^K \overset{\text{i.i.d.}}{\sim} q}\left[\log \frac{1}{K} \sum_{\ell=1}^{K} \frac{p(A, Z^\ell \mid \Pi, \Lambda)}{q(Z^\ell \mid A)}\right].$$

- **Tighter bound**: ELBO $= \mathcal{L}_1 \leq \mathcal{L}_K \leq \mathcal{L}_{K+1} \leq \log p(A \mid \Pi, \Lambda)$, converging to $\log p(A)$ as $K \to \infty$.
- **Better posterior approximation**: the K samples define an implicitly richer variational distribution that approaches the true posterior $p(Z \mid A)$ as K grows, beyond the mean-field family.

Inference: tightening the bound with the iw-ELBO

The latent variable lives on the simplex (Z constrained to a $Q-1$ manifold), so the mean-field posterior is a rough fit and the standard ELBO is a **loose** surrogate for $\log p(A \mid \Pi, \Lambda)$.

Starting from the exact marginal likelihood,

$$\log p(A \mid \Pi, \Lambda) = \log \mathbb{E}_{Z \sim q}\left[\frac{p(A, Z \mid \Pi, \Lambda)}{q(Z \mid A)}\right],$$

the importance-weighted ELBO (Burda et al., 2016) averages K samples *inside* the log:

$$\mathcal{L}_K = \mathbb{E}_{Z^1, \ldots, Z^K \overset{\text{i.i.d.}}{\sim} q}\left[\log \frac{1}{K} \sum_{\ell=1}^{K} \frac{p(A, Z^\ell \mid \Pi, \Lambda)}{q(Z^\ell \mid A)}\right].$$

- **Tighter bound**: ELBO $= \mathcal{L}_1 \leq \mathcal{L}_K \leq \mathcal{L}_{K+1} \leq \log p(A \mid \Pi, \Lambda)$, converging to $\log p(A)$ as $K \to \infty$.
- **Better posterior approximation**: the K samples define an implicitly richer variational distribution that approaches the true posterior $p(Z \mid A)$ as K grows, beyond the mean-field family.
- First use of the iw-ELBO for graph-based models.

The Data-set

We analysed the London Bike sharing dataset which:

- Records bike trips between $N = 779$ stations across London.
- The resulting graph comprises 224070 directed edges, where each entry (i, j) of the adjacency matrix indicates the number of trips from station i to station j
- Includes several station-level attributes, such as geographic coordinates (longitude and latitude) that we use only for interpretability of the results.

London bike sharing analysis

Entities are bike stations in London and interactions are the number of bike trips between them

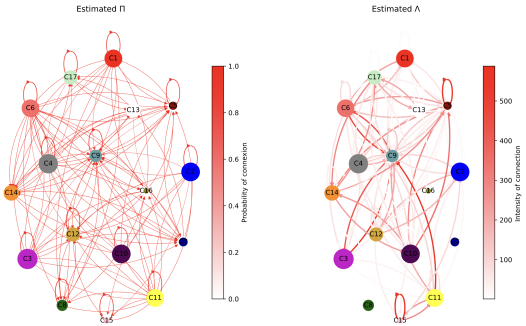


Figure 1: visualisation of the estimated parameters $\hat{\Pi}$ and $\hat{\Lambda}$ of Deep-ZLPBM

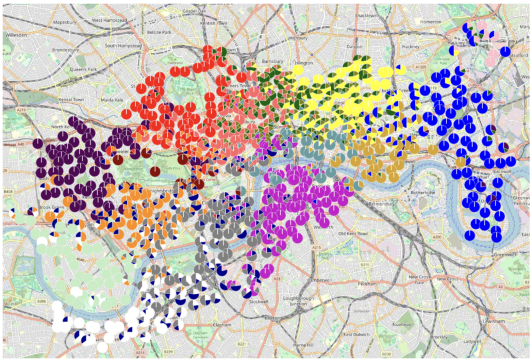


Figure 2: Visualisation of the retrieved clusters by Deep-ZLPBM on the London map

Deep-LPBM analysis

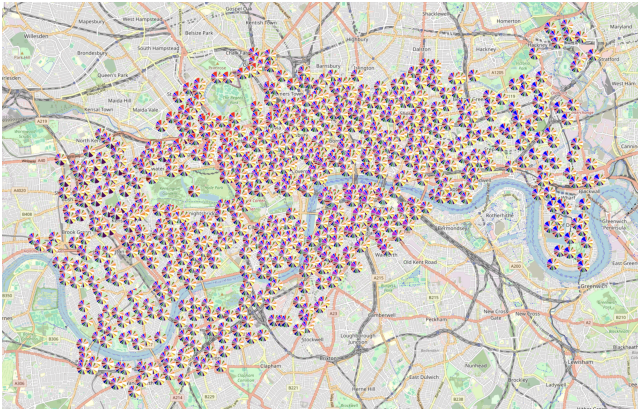


Figure 3: Visualisation of the retrieved clusters on Deep-LPBM in the London map street

Deep-LPBM analysis

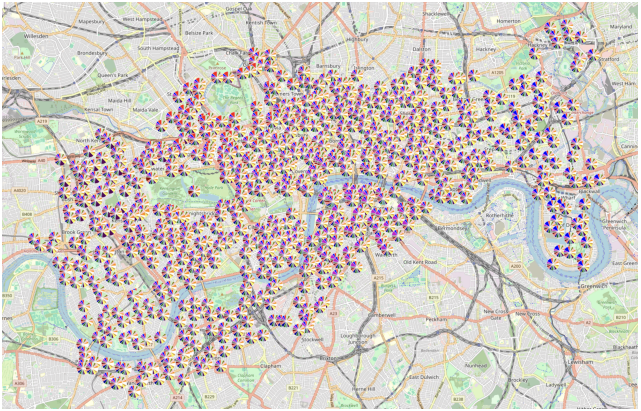


Figure 3: Visualisation of the retrieved clusters on Deep-LPBM in the London map street



Importance weighted directed graph variationalauto-encoder for block modelling of complex networks.

Niang, Bouveyron, Corneli, Latouche. **Journal of Statistical Analysis and Data Mining**, 2026.

Outline

The Deep Latent Position Block Model for the Clustering of Nodes in Multi-graphs

Deep zero-inflated Poisson latent position block model for valued networks

Large-Scale Graphon Learning with Topology-Aware Stochastic Optimization

Conclusion and perspectives

Graphons: a very general model for random graphs

A graphon is a **measurable** function $W : [0,1]^2 \to [0,1]$, with $W(u,v)$ the edge probability between positions u, v. Measurability is the *only* assumption.

It subsumes:

- **Erdős–Rényi** W constant
- **LPCM** W a function of the distance
- **SBM / Deep-LPBM** W piecewise constant

The price of this generality: estimation is hard.

Estimating a graphon: existing methods

Theoretical foundation. The weak regularity lemma (Lovász, 2012) guarantees that *any* graphon can be approximated by a **step function** (a $K \times K$ block matrix): for every K,

$$\|W - W_{\mathcal{P}}\|_{\square} \leq \frac{2}{\sqrt{\log K}} \|W\|_{L_2}.$$

- Block-model estimators fit the graphon **directly as an SBM** (a step function with K blocks), e.g. by Bayesian model averaging over SBMs (Latouche and Robin, 2016).

Estimating a graphon: existing methods

Theoretical foundation. The weak regularity lemma (Lovász, 2012) guarantees that *any* graphon can be approximated by a **step function** (a $K \times K$ block matrix): for every K,

$$\|W - W_{\mathcal{P}}\|_{\square} \leq \frac{2}{\sqrt{\log K}} \|W\|_{L_2}.$$

- Block-model estimators fit the graphon **directly as an SBM** (a step function with K blocks), e.g. by Bayesian model averaging over SBMs (Latouche and Robin, 2016).
- Histogram-based methods build on the same idea: they **sort the nodes** (by degree), then **fit a step function** on the reordered graph (Airoldi et al., 2013; Chan and Airoldi, 2014; Olhede and Wolfe, 2014). Fixed resolution K, cost $O(N^2)$.

Estimating a graphon: existing methods

Theoretical foundation. The weak regularity lemma (Lovász, 2012) guarantees that *any* graphon can be approximated by a **step function** (a $K \times K$ block matrix): for every K,

$$\|W - W_{\mathcal{P}}\|_{\square} \;\leq\; \frac{2}{\sqrt{\log K}}\,\|W\|_{L_2}.$$

- Block-model estimators fit the graphon **directly as an SBM** (a step function with K blocks), e.g. by Bayesian model averaging over SBMs (Latouche and Robin, 2016).
- Histogram-based methods build on the same idea: they **sort the nodes** (by degree), then **fit a step function** on the reordered graph (Airoldi et al., 2013; Chan and Airoldi, 2014; Olhede and Wolfe, 2014). Fixed resolution K, cost $O(N^2)$.
- Optimal-transport methods estimate W as a Gromov–Wasserstein barycentre, handling node alignment (Peyré, 2016; Xu et al., 2021), but scale as $O(N^3)$.

→ We propose SGVAE: a **deep** graphon learned end-to-end by maximum likelihood, with a topology-aware stochastic ELBO at $O(M + B)$ (M edges, B batch size).

Generative model: an implicit graphon decoder

We observe a symmetric and binary network $\mathbf{A}$, generated from an unknown graphon W

We use a neural network to estimate W:

$$A_{ij} \mid Z \sim \mathcal{B}\big(W_\theta(Z_i, Z_j)\big), \qquad Z_i \overset{\text{i.i.d.}}{\sim} \mathcal{U}([0,1]).$$

Generative model: an implicit graphon decoder

We observe a symmetric and binary network $\mathbf{A}$, generated from an unknown graphon W

We use a neural network to estimate W:

$$A_{ij} \mid Z \sim \mathcal{B}\big(W_\theta(Z_i, Z_j)\big), \qquad Z_i \overset{\text{i.i.d.}}{\sim} \mathcal{U}([0,1]).$$

- W_θ is a **SIREN** (sinusoidal implicit representation), symmetrised as $W_\theta(u,v) = \frac{1}{2}\big(f_\theta(u,v) + f_\theta(v,u)\big)$.

Generative model: an implicit graphon decoder

We observe a symmetric and binary network $\mathbf{A}$, generated from an unknown graphon W

We use a neural network to estimate W:

$$A_{ij} \mid Z \sim \mathcal{B}\big(W_\theta(Z_i, Z_j)\big), \qquad Z_i \overset{\text{i.i.d.}}{\sim} \mathcal{U}([0,1]).$$

- W_θ is a **SIREN** (sinusoidal implicit representation), symmetrised as $W_\theta(u,v) = \frac{1}{2}\big(f_\theta(u,v) + f_\theta(v,u)\big)$.
- No discretisation: the graphon is a continuous signal, not a fixed-resolution histogram.
- No Q: structure (communities, hubs, stars) is read off the learned surface, not imposed.

Generative model: an implicit graphon decoder

We observe a symmetric and binary network **A**, generated from an unknown graphon W

We use a neural network to estimate W:

$$A_{ij} \mid Z \sim \mathcal{B}\big(W_\theta(Z_i, Z_j)\big), \qquad Z_i \overset{\text{i.i.d.}}{\sim} \mathcal{U}([0,1]).$$

- W_θ is a **SIREN** (sinusoidal implicit representation), symmetrised as $W_\theta(u,v) = \frac{1}{2}\big(f_\theta(u,v) + f_\theta(v,u)\big)$.
- No discretisation: the graphon is a continuous signal, not a fixed-resolution histogram.
- No Q: structure (communities, hubs, stars) is read off the learned surface, not imposed.

The decoder alone defines the statistical model, exactly as in the previous chapters; here it is a neural network.

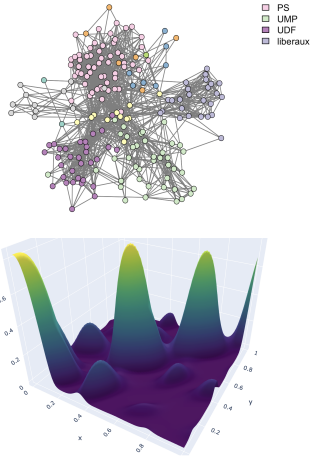


Figure 3: the french blogosphere network (top) and its estimated graphon (bottom) using SGVAE.

Inference: encoder, ELBO, and its bottleneck

Same amortised, mean-field recipe as before, with one change dictated by the latent space: positions live in $[0, 1]$, so the posterior is **Beta**, not Gaussian.

$$q_\phi(Z \mid A) = \prod_{i=1}^{N} \mathrm{Beta}(Z_i; \alpha_i, \beta_i), \qquad (\alpha_i, \beta_i) = \exp\big([g_\phi(A)]_i\big), \quad g_\phi \text{ a GCN.}$$

The ELBO keeps its two familiar terms,

$$\mathcal{L}(\theta, \phi) = \underbrace{\sum_{i<j} \mathbb{E}_{q_\phi}\big[\log p(A_{ij} \mid Z_i, Z_j, \theta)\big]}_{\text{reconstruction}} - \underbrace{\mathrm{KL}(q_\phi(\cdot \mid A) \,\|\, p(\cdot))}_{\text{regularisation}},$$

the KL in closed form (a sum of **Beta entropies**), the reconstruction estimated by Monte Carlo with the reparameterisation trick.

Inference: encoder, ELBO, and its bottleneck

Same amortised, mean-field recipe as before, with one change dictated by the latent space: positions live in $[0, 1]$, so the posterior is **Beta**, not Gaussian.

$$q_\phi(Z \mid A) = \prod_{i=1}^{N} \mathrm{Beta}(Z_i; \alpha_i, \beta_i), \qquad (\alpha_i, \beta_i) = \exp\big([g_\phi(A)]_i\big), \quad g_\phi \text{ a GCN.}$$

The ELBO keeps its two familiar terms,

$$\mathcal{L}(\theta, \phi) = \underbrace{\sum_{i<j} \mathbb{E}_{q_\phi}\big[\log p(A_{ij} \mid Z_i, Z_j, \theta)\big]}_{\text{reconstruction}} - \underbrace{\mathrm{KL}(q_\phi(\cdot \mid A) \,\|\, p(\cdot))}_{\text{regularisation}},$$

the KL in closed form (a sum of **Beta entropies**), the reconstruction estimated by Monte Carlo with the reparameterisation trick.

The reconstruction sums over **all pairs of nodes**, at cost $O(N^2)$: **prohibitive** for large networks. This quadratic cost is **common to all deep latent variable models** for graphs.

Topology-aware sampling

Idea. Avoid summing over all pairs: estimate the reconstruction by sampling pairs (i, j) and building a stochastic estimator of the ELBO.

But not uniformly. Large networks are sparse ($|\mathcal{E}| \ll |\mathcal{E}^c|$): a uniform draw returns **almost only non-edges**, giving batches of mostly zeros and a high-variance gradient.

Topology-aware sampling

Idea. Avoid summing over all pairs: estimate the reconstruction by sampling pairs (i, j) and building a stochastic estimator of the ELBO.

But not uniformly. Large networks are sparse ($|\mathcal{E}| \ll |\mathcal{E}^c|$): a uniform draw returns **almost only non-edges**, giving batches of mostly zeros and a high-variance gradient.

$\rightarrow$ **Sample edges and non-edges separately**, in a proportion set by ρ. Draw $Y \sim \mathcal{B}(\rho)$, then

$$(I, J) \sim \begin{cases} \mathrm{Unif}(\mathcal{E}) & \text{if } Y = 1 \text{ (an edge)}, \\ \mathrm{Unif}(\mathcal{E}^c) & \text{if } Y = 0 \text{ (a non-edge)}, \end{cases}$$

where $\mathcal{E} = \{(i, j) : i \neq j,\ A_{ij} = 1\}$, $\mathcal{E}^c$ the non-edges (both **oriented**: $\{i, j\}$ drawn as (i, j) or (j, i)).

Topology-aware sampling

Idea. Avoid summing over all pairs: estimate the reconstruction by sampling pairs (i, j) and building a stochastic estimator of the ELBO.

But not uniformly. Large networks are sparse ($|\mathcal{E}| \ll |\mathcal{E}^c|$): a uniform draw returns **almost only non-edges**, giving batches of mostly zeros and a high-variance gradient.

$\rightarrow$ **Sample edges and non-edges separately**, in a proportion set by ρ. Draw $Y \sim \mathcal{B}(\rho)$, then

$$(I, J) \sim \begin{cases} \mathrm{Unif}(\mathcal{E}) & \text{if } Y = 1 \text{ (an edge)}, \\ \mathrm{Unif}(\mathcal{E}^c) & \text{if } Y = 0 \text{ (a non-edge)}, \end{cases}$$

where $\mathcal{E} = \{(i, j) : i \neq j,\ A_{ij} = 1\}$, $\mathcal{E}^c$ the non-edges (both **oriented**: $\{i, j\}$ drawn as (i, j) or (j, i)).

Lemma 1 (pair law). $s(i, j) = \rho \dfrac{\mathbb{1}[(i, j) \in \mathcal{E}]}{|\mathcal{E}|} + (1 - \rho) \dfrac{\mathbb{1}[(i, j) \in \mathcal{E}^c]}{|\mathcal{E}^c|}$.

Lemma 2 (node marginal). $s(i) = \rho \dfrac{m_i}{|\mathcal{E}|} + (1 - \rho) \dfrac{N - m_i - 1}{|\mathcal{E}^c|}$ (m_i the degree of i).

An unbiased estimator of the ELBO

To marginalise the pair sampling over *all* oriented pairs, rewrite the reconstruction as if the graph were directed, at the price of a factor $\frac{1}{2}$:

$$\mathbb{E}_{Z\sim q_\phi}[\log p(A \mid Z, \theta)] = \sum_{i<j} \mathbb{E}[\cdot] = \tfrac{1}{2} \sum_{i\neq j} \mathbb{E}_{Z_i, Z_j \sim q_\phi}[\log p(A_{ij} \mid Z_i, Z_j, \theta)].$$

The ELBO then reads

$$\mathcal{L}(\theta, \phi) = \tfrac{1}{2} \sum_{i\neq j} \mathbb{E}_{Z_i, Z_j}[\log p(A_{ij} \mid Z_i, Z_j, \theta)] - \sum_i \mathbb{E}_{Z_i}[\log q_\phi(Z_i \mid A)].$$

For a pair $(I, J) \sim s$, define the single-pair estimator

$$G_{\theta,\phi}(I, J) := \underbrace{\frac{1}{2\, s(I, J)}\, \mathbb{E}_{Z_I, Z_J}[\log p(A_{IJ} \mid Z_I, Z_J, \theta)]}_{\text{reconstruction}} - \underbrace{\frac{1}{s(I)}\, \mathbb{E}_{Z_I}[\log q_\phi(Z_I \mid A)]}_{\text{regularisation}},$$

each term **reweighted by its sampling frequency** ($s(I, J)$ for pairs, $s(I)$ for nodes).

Unbiasedness and consistency

Theorem 1. Under the sampling law s, the mini-batch gradient $\nabla_{\theta,\phi}\widehat{\mathcal{L}}_B$ is an unbiased and strongly consistent estimator of $\nabla_{\theta,\phi}\mathcal{L}$:

$$\mathbb{E}_{(I,J)\sim s}\left[\nabla G_{\theta,\phi}(I,J)\right] = \nabla_{\theta,\phi}\mathcal{L}, \qquad \nabla_{\theta,\phi}\widehat{\mathcal{L}}_B \xrightarrow[B\to\infty]{\text{a.s.}} \nabla_{\theta,\phi}\mathcal{L}.$$

Proof sketch. Take $\mathbb{E}_{(I,J)\sim s}[G]$ term by term:

$$\text{reconstruction: } \sum_{i\neq j} s(i,j)\,\frac{1}{2\,s(i,j)}\,\mathbb{E}[\cdot] = \tfrac{1}{2}\sum_{i\neq j}\mathbb{E}[\cdot],$$

$$\text{regularisation: } \sum_{i\neq j} s(i,j)\,\frac{1}{s(i)}\,\mathbb{E}[\cdot] = \sum_{i}\frac{s(i)}{s(i)}\,\mathbb{E}[\cdot] = \sum_{i}\mathbb{E}[\cdot],$$

using $s(i) = \sum_{j\neq i} s(i,j)$ (Lemma 2). Hence $\mathbb{E}_{(I,J)\sim s}[G] = \mathcal{L}$. Since s **does not depend on** ϕ, gradient and expectation commute, so ∇G is unbiased; strong consistency then follows from the law of large numbers.

Unbiasedness and consistency

Theorem 1. Under the sampling law s, the mini-batch gradient $\nabla_{\theta,\phi}\widehat{\mathcal{L}}_B$ is an unbiased and strongly consistent estimator of $\nabla_{\theta,\phi}\mathcal{L}$:

$$\mathbb{E}_{(I,J)\sim s}\big[\nabla G_{\theta,\phi}(I,J)\big] = \nabla_{\theta,\phi}\mathcal{L}, \qquad \nabla_{\theta,\phi}\widehat{\mathcal{L}}_B \xrightarrow[B\to\infty]{\text{a.s.}} \nabla_{\theta,\phi}\mathcal{L}.$$

Proof sketch. Take $\mathbb{E}_{(I,J)\sim s}[G]$ term by term:

$$\text{reconstruction: } \sum_{i\neq j} s(i,j)\,\frac{1}{2\,s(i,j)}\,\mathbb{E}[\cdot] = \tfrac{1}{2}\sum_{i\neq j}\mathbb{E}[\cdot],$$

$$\text{regularisation: } \sum_{i\neq j} s(i,j)\,\frac{1}{s(i)}\,\mathbb{E}[\cdot] = \sum_{i}\frac{s(i)}{s(i)}\,\mathbb{E}[\cdot] = \sum_{i}\mathbb{E}[\cdot],$$

using $s(i) = \sum_{j\neq i} s(i,j)$ (Lemma 2). Hence $\mathbb{E}_{(I,J)\sim s}[G] = \mathcal{L}$. Since s **does not depend on** ϕ, gradient and expectation commute, so ∇G is unbiased; strong consistency then follows from the law of large numbers.

This is convergence in the **batch size** B for a fixed graph, not statistical consistency of the graphon estimate as $N \to \infty$.

Mini-batch optimisation and cost

Populate a batch $\mathcal{B}$ of B pairs (I_n, J_n) drawn i.i.d. from s, and take a gradient step on

$$\widehat{\mathcal{L}}_B(\theta, \phi) = \frac{1}{B} \sum_{n=1}^{B} G_{\theta,\phi}(I_n, J_n).$$

- **Per step:** $O(M + B)$, down from $O(N^2)$.
- **Over L epochs:** $O(L(M + B))$, near-linear in N on sparse graphs ($M \ll N^2$).
- The $O(M)$ comes only from the GNN encoder, which reads every edge once per pass; the sampled-pair SGD alone is $O(B)$. $O(M)$ is the information-theoretic floor for any method that must read the whole graph.

Mini-batch optimisation and cost

Populate a batch $\mathcal{B}$ of B pairs (I_n, J_n) drawn i.i.d. from s, and take a gradient step on

$$\widehat{\mathcal{L}}_B(\theta, \phi) = \frac{1}{B} \sum_{n=1}^{B} G_{\theta,\phi}(I_n, J_n).$$

- **Per step:** $O(M + B)$, down from $O(N^2)$.
- **Over L epochs:** $O(L(M + B))$, near-linear in N on sparse graphs ($M \ll N^2$).
- The $O(M)$ comes only from the GNN encoder, which reads every edge once per pass; the sampled-pair SGD alone is $O(B)$. $O(M)$ is the information-theoretic floor for any method that must read the whole graph.

Method	Res.-free	Complexity
SGVAE	✓	$O(L(M + B))$
SIGL	✓	$O(L(\frac{N^2}{h^2} + N^2))$
IGNR	✓	$O(L(\frac{N^3}{h} + \frac{N^3}{h^2}))$
GWB	✗	$O(L(\frac{N^3}{h} + \frac{N^3}{h^2}))$
SBA	✗	$O(N^2 + N^2 \log \frac{N^2}{h^2})$
USVT	✗	$O(N^3)$

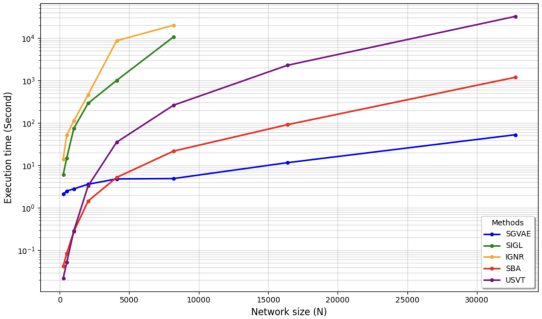
Graphon estimation accuracy

Benchmark on 15 synthetic graphons, Gromov–Wasserstein distance (**lower is better**; GW avoids the node-ordering ambiguity). Best in blue, second best in red.

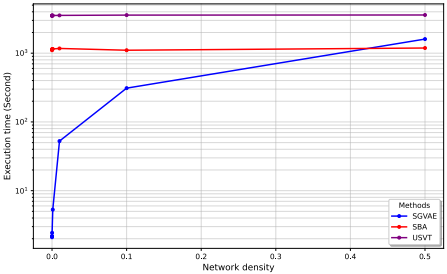
Method	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
SGVAE	.02	.02	.02	.02	.02	.02	.04	.04	.02	.01	.01	.00	.00	.01	.01
SIGL	.02	.02	.02	.02	.02	.02	.05	.05	.01	.01	.02	.01	.02	.06	.04
USVT	.02	.02	.04	.04	.04	.03	.05	.03	.06	.04	.06	.08	.09	.04	.05
IGNR	.05	.04	.05	.04	.06	.04	.07	.08	.06	.09	.09	.10	.45	.06	.04
GWB	.08	.06	.09	.10	.06	.08	.09	.04	.05	.07	.06	.08	.09	.16	.10
SBA	.20	.13	.18	.20	.13	.08	.11	.12	.15	.20	.20	.40	.40	.04	.03
LG	.20	.13	.18	.20	.13	.08	.11	.12	.15	.23	.23	.40	.40	.04	.03
OPTSPACE	.75	.20	.50	.19	.14	.18	.49	.10	.52	.70	.30	NA	NA	.91	.61

- Graphons **1–9** (smooth): SGVAE matches the best classical estimators (SIGL, USVT).
- Graphons **10–15** (blocks, discontinuities): the continuous SIREN stays accurate while SIGL and USVT degrade; others fail (NA = did not infer the network).

Empirical scaling



Runtime vs N, density fixed at 10^{-4}.



Runtime vs density, $N = 25{,}000$.

- SGVAE stays tractable across all scales, handling $N = 2^{20}$ (density 10^{-6}) in $\approx$ 5,800s, where USVT and SBA already fail at 2^{16}.
- Near-linear runtime growth with density, reflecting the $O(M)$ encoding cost; histogram and spectral baselines run out of memory or lag by orders of magnitude.

The latent geometry clusters for free

Graphon estimation is the goal; clustering comes as a by-product: K-means on the inferred positions $\hat{z}_i$, no clustering objective added. **ARI** (synthetic: mean over 10 runs; real: single run). Best in blue, second best in red.

Scenario	SGVAE	SBM*	Deep-LPBM	Louvain	InfoMap	LPCM	SIGL	FastGreedy
Assortative	.97	.99	.91	.92	.91	.83	.48	.90
Disassortative	.91	.92	.85	.00	.00	.00	.27	.00
Political Blogs	.59	.11	.10	.52	.43	.09	.37	.52
Amazon	.67	.18	.21	.42	.19	.07	.21	.48
Blogosphere	.70	.41	.32	.72	.78	.40	.18	.70

* SBM is an oracle reference on synthetic data only.

The latent geometry clusters for free

Graphon estimation is the goal; clustering comes as a by-product: K-means on the inferred positions $\hat{z}_i$, no clustering objective added. **ARI** (synthetic: mean over 10 runs; real: single run). Best in blue, second best in red.

Scenario	SGVAE	SBM*	Deep-LPBM	Louvain	InfoMap	LPCM	SIGL	FastGreedy
Assortative	.97	.99	.91	.92	.91	.83	.48	.90
Disassortative	.91	.92	.85	.00	.00	.00	.27	.00
Political Blogs	.59	.11	.10	.52	.43	.09	.37	.52
Amazon	.67	.18	.21	.42	.19	.07	.21	.48
Blogosphere	.70	.41	.32	.72	.78	.40	.18	.70

* SBM is an oracle reference on synthetic data only.

HAL science ouverte

Large-Scale Graphon Learning with Topology-Aware Stochastic Optimization. Niang, Bouveyron, Corneli, Latouche. **Preprint**, 2026.

Outline

The Deep Latent Position Block Model for the Clustering of Nodes in Multi-graphs

Deep zero-inflated Poisson latent position block model for valued networks

Large-Scale Graphon Learning with Topology-Aware Stochastic Optimization

Conclusion and perspectives

Conclusion: it is all in the decoder!

Across the three contributions, the encoder and the variational machinery stayed largely the same. What changed, and what carried the modelling power, was always the decoder.

- **Deep-LPBMM** (multi-view): one Bernoulli block matrix per view, sharing a single latent representation across layers.
- **Deep-ZLPBM** (valued, directed): a zero-inflated Poisson decoder with two block matrices, decoupling edge *existence* from edge *intensity*.
- **SGVAE / MGVAE** (scalable, non-parametric): the decoder becomes a measurable function, a graphon, learned end-to-end and trained at $O(L(M + B))$.

Takeaway: Designing better models therefore begins not with asking how to encode a network, but with asking what kind of network the decoder is able to create.

Perspectives

- **Integrate model selection**: infer the number of clusters *during* training rather than by refitting for each candidate.
- **Relax the shared representation** (Deep-LPBMM): give each view its own latent positions and learn how much each view contributes to a consensus geometry.
- **Scale the block models**: transfer the topology-aware estimator of SGVAE to Deep-LPBMM and Deep-ZLPBM, bringing them from $O(N^2)$ to $O(L(M+B))$.
- **Unify the three axes**: a single neural decoder handling valued, directed and multi-view networks at once, over a shared latent geometry.
- **Dynamic networks**: let the decoder evolve over time through a Markovian state, tracking structure that forms, merges or dissolves.

The final takeaway: by moving the modelling into a neural decoder, network analysis inherits the advances of deep learning on vectorial data.

Thank you for your attention

Seydina Ousmane NIANG

seydina-ousmane.niang@inria.fr

Supervision: C. Bouveyron, P. Latouche, & M. Corneli

Contributions and publications

My publications

- **Deep-LPBMM.** Niang, Bouveyron, Corneli, Latouche, Boutin. *The Deep Latent Position Block Model for the Clustering of Nodes in Multi-graphs.* Journal of Classification, 2026.

- **Deep-ZLPBM.** Niang, Bouveyron, Corneli, Latouche. *Importance-Weighted Directed Graph VAE for Block Modelling of Complex Networks.* Statistical Analysis and Data Mining,,2026.

- **SGVAE / MGVAE.** Niang, Bouveyron, Latouche, Corneli. *Large-Scale Graphon Learning with Topology-Aware Stochastic Optimization.* In preparation.



Deep-LPBMM



Deep-ZLPBM



SGVAE / MGVAE