Efficient Learning on Large Graphs using a Densifying Regularity Lemma

Jonathan Kouchly[*]
Technion – Israel Institute of Technology

Ben Finkelshtein[*]
University of Oxford

Michael Bronstein
University of Oxford / AITHYRA

Ron Levie
Technion – Israel Institute of Technology

Abstract

Learning on large graphs presents significant challenges, with traditional Message Passing Neural Networks suffering from computational and memory costs scaling linearly with the number of edges. We introduce the Intersecting Block Graph (IBG), a low-rank factorization of large directed graphs based on combinations of intersecting bipartite components, each consisting of a pair of communities, for source and target nodes. By giving less weight to non-edges, we show how an IBG can efficiently approximate any graph, sparse or dense. Specifically, we prove a constructive version of the weak regularity lemma: for any chosen accuracy, every graph can be approximated by a dense IBG whose rank depends only on that accuracy. This improves over prior versions of the lemma, where the rank depended on the number of nodes for sparse graphs. We then introduce a graph neural network architecture operating on the IBG representation of the graph and demonstrating competitive performance on node classification, spatio-temporal graph analysis, and knowledge graph completion, while having memory and computational complexity linear in the number of nodes rather than edges.

1 Introduction

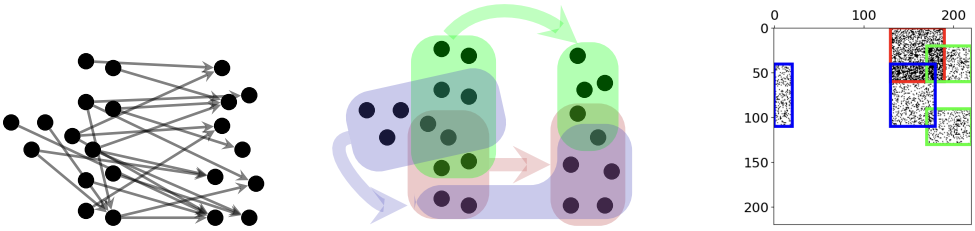
Graphs are a powerful representation for structured data, with applications spanning social networks [1, 2], biological systems [3], traffic modeling [4], and knowledge graphs [5], to name a few. As graph sizes continue to grow in application, learning on such large-scale graphs presents computational and memory challenges. Traditional Message Passing Neural Networks (MPNNs), which form the backbone of most graph signal processing architectures, scale their computational and memory requirements linearly with the *number of edges*. This edge-dependence limits their scalability in some situations, e.g., when processing social networks that can typically have $10^8 \sim 10^9$ nodes and $10^2 \sim 10^3$ as many edges [6].

Several strategies, called *graph reduction methods*, have been proposed to alleviate these challenges. These include *graph sparsification*, where a smaller graph is randomly sampled from the large graph [1, 2]; *graph condensation*, where a new small graph is created [7, 8], representing structures in the large graph; and *graph coarsening*, where sets of nodes are grouped into super nodes [9, 10]. However, with the exception of graph sparsification, graph reduction methods typically do not address the problem of processing a graph that is too large to fit at once in memory (e.g., on the GPU).

Recently, Finkelshtein et al. [11] proposed using a low-rank approximation of the graph, called *Intersecting Community Graph* (ICG), instead of the graph itself, for processing the data. When training a model on the ICG, the computational complexity is reduced from linear in the number of edges (as in MPNNs) to linear in the *number of nodes*. However, ICG has a number of limitations. First, the ICG approximation quality degrades with the sparsity of the graph, making ICG appropriate

[*]Equal contribution. Correspondence to: kjonathan@campus.technion.ac.il

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: New Perspectives in Advancing Graph Machine Learning.



(a) A directed graph. (b) Approximating 3-IBG. (c) The intersecting blocks of the 3-IBG.

Figure 1: (**a**) Directed graph sampled from a stochastic block model (SBM); (**b**) The approximating 3-IBG, where source and target community pairs are represented by the same color; (**c**) Adjacency matrix of a graph sampled from the same SBM, with the 3-IBG overlaid on the adjacency matrix.

mainly for graphs that are both large and dense. Secondly, ICGs can only approximate undirected graphs. In this paper, we propose a non-trivial extension of the ICG method, called *Intersecting Blocks Graph (IBG)*, addressing the aforementioned limitaitons and more.

Our contribution. We introduce a new procedure for approximating general directed graphs G with adjacency matrices $\boldsymbol{A} \in \{0,1\}^{N\times N}$ by low-rank matrices $\boldsymbol{C}$ that have a special interpretation. The approximating graph consists of a set of overlapping bipartite components. Namely, there is a set of $K \ll N$ pairs of node communities $(\mathcal{U}_i, \mathcal{V}_i)$, $i = 1, \ldots, K$, and each pair defines a weighted bipartite component, in which edges connect each node of $\mathcal{U}_i$ to each node of $\mathcal{V}_i$ with some weight r_i (that can be negative). The graph $\boldsymbol{C}$ is defined as the sum of all of these components, called *blocks* or *directed communities*, where the different communities can overlap. We demonstrate how processing $\boldsymbol{C}$ instead of $\boldsymbol{A}$ leads to models that solve downstream tasks in linear time and complexity with respect to the number of nodes, as opposed to standard MPNNs that are linear in the number of edges.

To fit $\boldsymbol{C}$ to $\boldsymbol{A}$, we consider a loss $L_{\boldsymbol{A}}(\boldsymbol{C})$ defined as a *weighted norm* of $\boldsymbol{A} - \boldsymbol{C}$, i.e., a standard norm weighted element-wise by a weight matrix $\boldsymbol{Q} \in (0,\infty)^{N\times N}$. The goal of using weights is to balance the contributions of edges and non-edges. The weight matrix $\boldsymbol{Q}$ is chosen adaptively, depending on the target adjacency matrix $\boldsymbol{A}$. We consider a *weighted cut norm* as the approximation metric. The cut norm is a well-established graph similarity measure that we discuss in Section 3, and Appendices B and C. Notably, The Weak Regularity Lemma (WRL) asserts that one can approximate any graph with E edges and N nodes up to error ϵ w.r.t. the cut metric by a low-rank graph consisting of $N/(\sqrt{E}\epsilon^2)$ intersecting communities. However, computing the cut metric is NP-hard, which prohibits explicitly optimizing it. To solve this, we propose a new version of the WRL, showing how to approximate any graph with a low rank approximation efficiently and tractably. Notably, our methods achieves an approximation bound that is independent of the sparsity of the graph, whereas the bound for ICG increased with the size of the of the graph for sparse graph.

We emphasize that this independence of the number of communities on the sparsity level is not merely an artifact of renormalizing the loss to artificially facilitate the desired error bound. Rather, the loss is deliberately designed to promote denseness when approximating graphs. The ability to efficiently densify a given graph can improve downstream tasks, as the densified version $\boldsymbol{C}^*$ of the graph $\boldsymbol{A}$ can often strengthen the connectivity patterns of the graph. We call the approximating low-rank graph IBG. For comparison with the predecessor of our method, ICG [11], see Appendices A and J.

2 Basic definitions and notations

We denote matrices by boldface uppercase letters, e.g., $\boldsymbol{D}$, vectors by boldface lowercase $\boldsymbol{d}$, and their scalar entries by the same lowercase letter d_i with subscript for the index.

Graph signals. We consider *directed* graphs G with sets of N nodes $\mathcal{V} = [N] = \{1, \ldots, N\}$, E edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, adjacency matrix $\boldsymbol{A} = (a_{i,j})_{i,j=1}^{N} \in \{0,1\}^{N\times N}$, and node feature matrix $\boldsymbol{X} = (x_{i,j})_{i,j=1}^{N,D} \in [-1,1]^{N\times D}$, called the *signal*. We follow the graph signal processing convention and represent the data as *graph-signals* $G = (\boldsymbol{A}, \boldsymbol{X})$. We denote the j-th column of the matrix $\boldsymbol{Q}$ by $\boldsymbol{Q}_{:,j}$, the i-th row by $\boldsymbol{Q}_{i,:}$, and $\mathrm{diag}(\boldsymbol{r}) \in \mathbb{R}^{K\times K}$ the diagonal matrix with values $\boldsymbol{r} \in \mathbb{R}^K$.

Frobenius norm. The *weighted Frobenius norm* of a square matrix $\boldsymbol{D} \in \mathbb{R}^{N\times N}$ with respect to the *weight* $\boldsymbol{Q} \in (0,\infty)^{N\times N}$ is defined to be $\|\boldsymbol{D}\|_{\mathrm{F};\boldsymbol{Q}} := \left(\frac{1}{\sum_{i,j=1}^{N} q_{i,j}} \sum_{i,j=1}^{N} d_{i,j}^2 q_{i,j}\right)^{1/2}$. Denote

$\|\boldsymbol{D}\|_{\mathrm{F}} := \|\boldsymbol{D}\|_{\mathrm{F};\mathbf{1}}$, where $\mathbf{1}$ is the all-1 matrix. The Frobenius norm of a signal $\boldsymbol{Y} \in \mathbb{R}^{N\times D}$ is defined by $\|\boldsymbol{Y}\|_{\mathrm{F}} := \sqrt{\frac{1}{ND}\sum_{j=1}^{D}\sum_{i=1}^{N} y_{i,j}^2}$. The *weighted Frobenius norm with weights* $\alpha, \beta > 0$ of a matrix-signal $(\boldsymbol{D}, \boldsymbol{Y})$ is defined by $\|(\boldsymbol{D}, \boldsymbol{Y})\|_{\mathrm{F};\boldsymbol{Q}} = \|(\boldsymbol{D}, \boldsymbol{Y})\|_{\mathrm{F};\boldsymbol{Q},\alpha,\beta} := \sqrt{\alpha \|\boldsymbol{D}\|_{\mathrm{F};\boldsymbol{Q}}^2 + \beta \|\boldsymbol{Y}\|_{\mathrm{F}}^2}$.

3 Weighted graph similarity measures

Weighted cut-metric. The *cut-metric* is a graph similarity measure based on the *cut-norm*. Below, we define it for graphs of the same size.

Definition 3.1. *The* weighted matrix cut-norm *of* $\boldsymbol{D} \in \mathbb{R}^{N\times N}$ *with weights* $\boldsymbol{Q} \in (0,\infty)^{N\times N}$*, and the* signal cut-norm *of* $\boldsymbol{Y} \in \mathbb{R}^{N\times D}$*, are defined respectively as*

$$\|\boldsymbol{D}\|_{\square;\boldsymbol{Q}} = \frac{1}{\sum_{i,j} q_{i,j}} \max_{\mathcal{U},\mathcal{V}\subset[N]} \Big|\sum_{i\in\mathcal{U}}\sum_{j\in\mathcal{V}} d_{i,j}q_{i,j}\Big| \qquad \textit{and} \qquad \|\boldsymbol{Y}\|_{\square} = \frac{1}{DN}\sum_{j=1}^{D}\max_{\mathcal{U}\subset[N]}\Big|\sum_{i\in\mathcal{U}} y_{i,j}\Big|$$

The weighted matrix-signal cut-norm *of* $(\boldsymbol{D}, \boldsymbol{Y})$*, with weights* $\alpha, \beta > 0$*, is then*

$$\|(\boldsymbol{D}, \boldsymbol{Y})\|_{\square;\boldsymbol{Q}} = \|(\boldsymbol{D}, \boldsymbol{Y})\|_{\square;\boldsymbol{Q},\alpha,\beta} := \alpha \|\boldsymbol{D}\|_{\square;\boldsymbol{Q}} + \beta \|\boldsymbol{Y}\|_{\square}\,. \tag{1}$$

Densifying cut similarity. In sparse graphs, such as those in link prediction and knowledge graph completion [12, 13], the number of non-edges far exceeds that of edges. This imbalance causes the cut-metric to be dominated by non-edges unless they are properly weighted. We believe this effect contributed to the weaker performance of the low-rank ICG approximation in [11], which relied on an unweighted cut-metric, i.e. $\boldsymbol{Q} = \mathbf{1}$ and $\|\boldsymbol{D}\|_{\square} := (N^2/E)\,\|\boldsymbol{D}\|_{\square;\mathbf{1}}$.

To address this, we introduce the *densifying cut similarity*, a weighted version of the cut-metric that down-weights non-edges. We define a matrix $\boldsymbol{Q}$ that assigns a weight e to non-edges and 1 to edges, where e is calibrated by a parameter $\Gamma > 0$ that balances the contribution of edges and non-edges. See Appendix C for further motivation of the definition.

Definition 3.2. *Let* $\boldsymbol{A} \in \{0,1\}^{N\times N}$ *be an unweighted adjacency matrix, and* $\Gamma > 0$*. The* densifying cut similarity *between the target* $\boldsymbol{A}$ *and any adjacency matrix* $\boldsymbol{B} \in \mathbb{R}^{N\times N}$ *is defined to be*

$$\sigma_{\square}(\boldsymbol{A}||\boldsymbol{B}) = \sigma_{\square;\Gamma}(\boldsymbol{A}||\boldsymbol{B}) := (1+\Gamma)\,\|\boldsymbol{A}-\boldsymbol{B}\|_{\square;\boldsymbol{Q}_{\boldsymbol{A}}}\,,$$

where the weight matrix $\boldsymbol{Q}_{\boldsymbol{A}}$ *is*

$$\boldsymbol{Q}_{\boldsymbol{A}} = \boldsymbol{Q}_{\boldsymbol{A},\Gamma} := e_{E,\Gamma}\mathbf{1} + (1-e_{E,\Gamma})\boldsymbol{A}, \quad \textit{with } e_{E,\Gamma} = \frac{\Gamma E/N^2}{1-(E/N^2)}. \tag{2}$$

Given $\alpha, \beta > 0$ *such that* $\alpha+\beta = 1$*, the* densifying cut similarity *between the target graph-signal* $(\boldsymbol{A}, \boldsymbol{X})$ *and the graph-signal* $(\boldsymbol{A}', \boldsymbol{X}')$ *is defined to be*

$$\sigma_{\square}\big((\boldsymbol{A}, \boldsymbol{X})||(\boldsymbol{A}', \boldsymbol{X}')\big) = \sigma_{\square;\alpha,\beta,\Gamma}\big((\boldsymbol{A}, \boldsymbol{X})||(\boldsymbol{A}', \boldsymbol{X}')\big) := \alpha\sigma_{\square;\Gamma}(\boldsymbol{A}||\boldsymbol{B}) + \beta\,\|\boldsymbol{X}-\boldsymbol{X}'\|_{\square}\,.$$

4 Approximations by intersecting blocks

4.1 Intersecting block graphs

For any subset of nodes $\mathcal{U} \subset [N]$, the indicator function $\mathbb{1}_{\mathcal{U}}$ is defined as $\mathbb{1}_{\mathcal{U}}(i) = 1$ if $i \in \mathcal{U}$ and 0 otherwise. As explained above, we treat $\mathbb{1}_{\mathcal{U}}$ as a vector in $\mathbb{R}^N$. Denote by χ the set of all such indicator functions. We define an *Intersecting Block Graph (*IBG*)* with K classes (K-IBG) as a low-rank graph-signal $(\boldsymbol{C}, \boldsymbol{P})$ with adjacency matrix and signals given respectively by

$$\boldsymbol{C} = \sum_{j=1}^{K} r_j\, \mathbb{1}_{\mathcal{U}_j}\mathbb{1}_{\mathcal{V}_j}^{\top}, \quad \boldsymbol{P} = \sum_{j=1}^{K} \mathbb{1}_{\mathcal{U}_j}\boldsymbol{f}_j^{\top} + \mathbb{1}_{\mathcal{V}_j}\boldsymbol{b}_j^{\top}$$

where $r_j \in \mathbb{R}$, $\boldsymbol{f}_j, \boldsymbol{b}_j \in \mathbb{R}^D$, and $\mathcal{U}_j, \mathcal{V}_j \subset [N]$. Next, we relax the $\{0,1\}$-valued hard indicator functions $\mathbb{1}_{\mathcal{U}}, \mathbb{1}_{\mathcal{V}}$ to *soft affiliation functions* with values in $\mathbb{R}$, as defined next, to allow continuously optimizing IBGs.

Definition 4.1. *A set* $\mathcal{Q}$ *of vectors* $\boldsymbol{u} : [N] \to \mathbb{R}$ *that contains* χ *is called a* soft affiliation model*.*

Definition 4.2. *Let* $d \in \mathbb{N}$*, and let* $\mathcal{Q}$ *be a soft affiliation model. We define* $[\mathcal{Q}] \subset \mathbb{R}^{N\times N} \times \mathbb{R}^{N\times D}$ *to be the set of all elements of the form* $(r\boldsymbol{u}\boldsymbol{v}^{\top}, \boldsymbol{u}\boldsymbol{f}^{\top} + \boldsymbol{v}\boldsymbol{b}^{\top})$*, with* $\boldsymbol{u}, \boldsymbol{v} \in \mathcal{Q}$*,* $r \in \mathbb{R}$ *and* $\boldsymbol{f}, \boldsymbol{b} \in \mathbb{R}^D$*. We call* $[\mathcal{Q}]$ *the* soft rank-1 intersecting block graph (IBG) model *corresponding to* $\mathcal{Q}$*. Given* $K \in \mathbb{N}$*, the subset* $[\mathcal{Q}]_K$ *of* $\mathbb{R}^{N\times N} \times \mathbb{R}^{N\times D}$ *of all linear combinations of* K *elements of* $[\mathcal{Q}]$ *is called the* soft rank-K IBG model *corresponding to* $\mathcal{Q}$*.*

In matrix form, an IBG $(\boldsymbol{C}, \boldsymbol{P}) \in \mathbb{R}^{N\times N} \times \mathbb{R}^{N\times D}$ in $[\mathcal{Q}]_K$ can be written as

$$\boldsymbol{C} = \boldsymbol{U}\,\mathrm{diag}(\boldsymbol{r})\boldsymbol{V}^\top \quad \text{and} \quad \boldsymbol{P} = \boldsymbol{U}\boldsymbol{F} + \boldsymbol{V}\boldsymbol{B} \tag{3}$$

via the *target community affiliation matrix* $\boldsymbol{U} \in \mathbb{R}^{N\times K}$, *source community affiliation matrix* $\boldsymbol{V} \in \mathbb{R}^{N\times K}$, *community magnitude vector* $\boldsymbol{r} \in \mathbb{R}^K$, *target community feature matrix* $\boldsymbol{F} \in \mathbb{R}^{K\times D}$ and *source community feature matrix* $\boldsymbol{B} \in \mathbb{R}^{K\times D}$.

4.2 The densifying regularity lemma

Directly minimizing the cut metric is computationally difficult since it involves a maximization step, making the optimization a min-max problem. While [14] proposed an algorithm that addresses this issue, it comes with an impractical runtime of $Ne^{\mathcal{O}(K)}$. To overcome this, we introduce a new semi-constructive version of the WRL for intersecting blocks. The approach is termed semi-constructive because it formulates the approximating graph as the solution to an "easy-to-solve" optimization problem that can be efficiently handled using standard gradient descent techniques.

The theorem generalizes the semi-constructive WRL based on intersecting communities of [11] in three main ways: (1) extending the theorem to directed graphs and the densifying graph similarity, instead of undirected graphs and the cut norm, (2) introducing a certificate for testing that the high probability event in which the cut similarity error is small occurred, and the key novelty of this theorem – (3) providing a bound that is independent of the graph size for both sparse and dense graphs, whereas the bound in [11] depended on the graph size for sparse graphs.

Theorem 4.1. *Let $(\boldsymbol{A}, \boldsymbol{X})$ be a graph-signal, $K \in \mathbb{N}$, $\delta > 0$, and let $\mathcal{Q}$ be a soft indicators model. Let $\alpha, \beta > 0$ such that $\alpha+\beta = 1$. Let $\Gamma > 0$ and let $\boldsymbol{Q}_{\boldsymbol{A}}$ be the weight matrix defined in Definition 3.2. Let $R \geq 1$ such that $K/R \in \mathbb{N}$. For every $k \in \mathbb{N}$, let*

$$\eta_k = (1+\delta) \min_{(\boldsymbol{C},\boldsymbol{P})\in[\mathcal{Q}]_k} \|(\boldsymbol{A}, \boldsymbol{X}) - (\boldsymbol{C}, \boldsymbol{P})\|^2_{\mathrm{F};\boldsymbol{Q}_{\boldsymbol{A}},\alpha(1+\Gamma),\beta}\,.$$

Then,

1. *For every $m \in \mathbb{N}$, any* IBG $(\boldsymbol{C}^*, \boldsymbol{P}^*) \in [\mathcal{Q}]_m$ *that gives a close-to-best weighted Frobenius approximation of $(\boldsymbol{A}, \boldsymbol{X})$ in the sense that*

$$\|(\boldsymbol{A}, \boldsymbol{X}) - (\boldsymbol{C}^*, \boldsymbol{P}^*)\|^2_{\mathrm{F};\boldsymbol{Q}_{\boldsymbol{A}},\alpha(1+\Gamma),\beta} \leq \eta_m, \tag{4}$$

   also satisfies

$$\sigma_{\square;\alpha,\beta,\Gamma}\big((\boldsymbol{A}, \boldsymbol{X})\|(\boldsymbol{C}^*, \boldsymbol{P}^*)\big) \leq (\sqrt{\alpha(1+\Gamma)} + \sqrt{\beta})\sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}}. \tag{5}$$

2. *If m is uniformly randomly sampled from $[K]$, then in probability $1 - \frac{1}{R}$,*

$$\sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}} \leq \sqrt{\delta + \frac{R(1+\delta)}{K}}. \tag{6}$$

   Specifically, in probability $1 - \frac{1}{R}$, any $(\boldsymbol{C}^*, \boldsymbol{P}^*) \in [\mathcal{Q}]_m$ which satisfies (4), also satisfies

$$\sigma_{\square,\alpha,\beta}\big((\boldsymbol{A}, \boldsymbol{X}) - (\boldsymbol{C}^*, \boldsymbol{P}^*)\big) \leq \sqrt{2+\Gamma}\sqrt{\delta + \frac{R(1+\delta)}{K}}. \tag{7}$$

4.3 Fitting intersecting blocks using gradient descent

In this section, we propose an efficient computation for fitting IBGs to directed graphs based on Theorem 4.1 (minimizing the left-hand-side of (4) via gradient descent). As the soft affiliation model, we consider all vectors in $[0,1]^N$. In the notations of (3), we optimize the parameters $\boldsymbol{U}, \boldsymbol{V} \in [0,1]^{N\times K}$, $\boldsymbol{r} \in \mathbb{R}^K$ and $\boldsymbol{F}, \boldsymbol{B} \in \mathbb{R}^{K\times D}$ to minimize the weighted Frobenius norm

$$L(\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{r}, \boldsymbol{F}, \boldsymbol{B}) = \alpha(1+\Gamma)\left\|\boldsymbol{A} - \boldsymbol{U}\,\mathrm{diag}(\boldsymbol{r})\boldsymbol{V}^\top\right\|^2_{\mathrm{F};\boldsymbol{Q}_{\boldsymbol{A}}} + \beta\left\|\boldsymbol{X} - \boldsymbol{U}\boldsymbol{F} - \boldsymbol{V}\boldsymbol{B}\right\|^2_{\mathrm{F}}. \tag{8}$$

In practice, we implement $\boldsymbol{U}, \boldsymbol{V} \in [0,1]^{N\times K}$ by applying a sigmoid activation function to learned matrices $\boldsymbol{U}', \boldsymbol{V}' \in \mathbb{R}^{N\times K}$, setting $\boldsymbol{U} = \mathrm{Sigmoid}(\boldsymbol{U}')$ and $\boldsymbol{V} = \mathrm{Sigmoid}(\boldsymbol{V}')$.

Optimizing (8) naïvely requires $\mathcal{O}(N^2)$ operations, as the matrix $\boldsymbol{A} - \boldsymbol{U}\,\mathrm{diag}(\boldsymbol{r})\boldsymbol{V}^\top \in \mathbb{R}^{N\times N}$ is not sparse nor low-rank. However, we can exploit the sparsity of $\boldsymbol{A}$ and the low-rank structure of $\boldsymbol{U}\,\mathrm{diag}(\boldsymbol{r})\boldsymbol{V}^\top$ separately to enable an efficient computation with time and space complexities of $\mathcal{O}(K^2N + KE)$ and $\mathcal{O}(KN + E)$, respectively.

Proposition 4.2. *Let $\boldsymbol{A} = (a_{i,j})_{i,j=1}^{N}$ be an adjacency matrix of an unweighted graph with E edges. The graph part of the sparse Frobenius loss (8) can be written as*

$$\left\|\boldsymbol{A} - \boldsymbol{U}\,\mathrm{diag}(\boldsymbol{r})\boldsymbol{V}^\top\right\|^2_{\mathrm{F};\boldsymbol{Q_A}} = \|A\|^2_{\mathrm{F};\boldsymbol{Q_A}} + \frac{e_{E,\Gamma}}{(1+\Gamma)\,E}\,\mathrm{Tr}\left((\boldsymbol{V}^\top\boldsymbol{V})\,\mathrm{diag}(\boldsymbol{r})(\boldsymbol{U}^\top\boldsymbol{U})\,\mathrm{diag}(\boldsymbol{r})\right)$$

$$- \frac{2}{(1+\Gamma)\,E}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}(i)}\boldsymbol{U}_{i,:}\,\mathrm{diag}(\boldsymbol{r})\left(\boldsymbol{V}^\top\right)_{:,j}a_{i,j} + \frac{1-e_{E,\Gamma}}{(1+\Gamma)\,E}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}(i)}(\boldsymbol{U}_{i,:}\,\mathrm{diag}(\boldsymbol{r})\left(\boldsymbol{V}^\top\right)_{:,j})^2$$

where $\boldsymbol{Q_A}$ and $e_{E,\Gamma}$ are defined in (2). Computing the right-hand-side and its gradients with respect to $\boldsymbol{U}$, $\boldsymbol{V}$ and $\boldsymbol{r}$ has a time complexity of $\mathcal{O}(K^2N + KE)$, and a space complexity of $\mathcal{O}(KN + E)$.

The parametres of the IBG are optimized via gradient descent on (8), restructured as Proposition 4.2.

4.4 The learning pipeline with IBGs

When learning on graphs using IBGs, the first step is fitting an IBG to the given graph. This is done once in $\mathcal{O}(E)$ time and memory complexity. The second step is solving the task, e.g., node classification, or spatio-temporal prediction. This step typically involves an extensive architecture and hyperparameter search. In our pipeline, the neural network processes the IBG representation of the data, instead of the standard representation of the graph. This improves the processing time and memory complexity from $\mathcal{O}(E)$ to $\mathcal{O}(N)$. Thus, searching through $S \in \mathbb{N}$ hyperparameter configurations takes $\mathcal{O}(SN)$ time, while learning directly on the graph would take $\mathcal{O}(SE)$.

Initialization of IBG. In Appendix G we explain how to use a low-rank SVD of the graph to initialize a rank K IBG, before the IBG fitting process. We also propose a randomized SVD algorithm for approximating the SVD while only loading a fraction of the graph into memory (Appendix G.2).

Subgraph SGD for fitting IBGs to large graphs. Fitting an IBG to a graph requires $\mathcal{O}(E)$ memory complexity, which may exceed the GPU capacity in some situations. To solve this, in Appendix H we propose a sampling approach for optimizing the IBG, which reduces the memory complexity, allowing fitting IBGs to large graphs on hardware with limited memory.

5 Processing IBGs with neural networks

Graph signal processing with IBG. In this section we provide a paradigm for learning on IBGs, which runs in $\mathcal{O}(NK)$ operations per layer, which is often faster than the $\mathcal{O}(E)$ complexity of MPNNs. Let $\boldsymbol{U}, \boldsymbol{V} \in \mathbb{R}^{N\times K}$ be target and source community affiliation matrices. We call $\mathbb{R}^{N\times D}$ the *node space* and $\mathbb{R}^{K\times D}$ the *community space*. We use the following operations to process signals:

- The mappings $\boldsymbol{F} \mapsto \boldsymbol{U}\boldsymbol{F}, \boldsymbol{B} \mapsto \boldsymbol{V}\boldsymbol{B}$ from the *community space* to the *node space*, in $O(NKD)$.
- Any function of $\boldsymbol{F}$ and $\boldsymbol{B}$ in the *community space* (e.g., an MLP) in $O(K^2D^2)$ (or less).
- Any function that operates on node features in the *node space*, in $O(ND^2)$ operations (or less).

Deep IBG Neural Networks. We propose an IBG-based deep architecture which takes $O(D(NK + KD + ND))$ operations at each layer. Our IBG neural network (IBG-NN) is defined as follows. Let $D^{(\ell)}$ denote the dimension of the node features at layer ℓ, and set the initial node representations as $\boldsymbol{H}^{(0)} = \boldsymbol{X}$. Then, for layers $0 \le \ell \le L-1$, the node features are defined by

$$\boldsymbol{H}_s^{(\ell+1)} = \sigma\left(\Theta_1^s\left(\boldsymbol{H}_s^{(\ell)}\right) + \Theta_2^s\left(\boldsymbol{V}\boldsymbol{B}^{(\ell)}\right)\right), \quad \boldsymbol{H}_t^{(\ell+1)} = \sigma\left(\Theta_1^t\left(\boldsymbol{H}_t^{(\ell)}\right) + \Theta_2^t\left(\boldsymbol{U}\boldsymbol{F}^{(\ell)}\right)\right).$$

The final representation is taken as $\boldsymbol{H}^{(L)} = \boldsymbol{H}_s^{(L)} + \boldsymbol{H}_t^{(L)}$, where Θ_1 and Θ_2 are MLPs or multiple layers of deepsets, $\boldsymbol{F}^{(\ell)}, \boldsymbol{B}^{(\ell)} \in \mathbb{R}^{K\times D^{(\ell)}}$ are taken directly as trainable parameters, and σ is a non-linearity. The final representations $\boldsymbol{H}^{(L)}$ can be used for predicting node-level properties. See Appendix L for more details on IBG-NNs, and an extension of IBG-NNs for spatio-temporal graphs.

6 Experiments

6.1 The efficient run-time of IBG-NNs

Setup. We compare the forward pass runtimes of IBG-NN and DirGNN [20], a simple and efficient MPNN for directed graphs, on *Erdős-Rényi* $\mathrm{ER}(n, p = 0.5)$ graphs with up to 7k nodes. We use 128 node features per node. Both models use hidden dimensions of 128, and 3 layers.

Results. Figure 2 shows that the runtime of IBG-NN exhibits a square root relationship compared to DirGNN. This matches expectations, as their respective time complexities are $\mathcal{O}(N)$ and $\mathcal{O}(E)$.

Table 1: Results on graph densification benchmarks; top models colored First, Second, Third.

Condensation ratio	Flickr 0.5%	Flickr 1%	Flickr 100%	Reddit 0.1%	Reddit 0.2%	Reddit 100%
Coarsening [15]	44.5 ± 0.1	44.6 ± 0.1	47.2 ± 0.1	42.8 ± 0.8	47.4 ± 0.9	93.9 ± 0.0
Random [15]	44.0 ± 0.4	44.6 ± 0.2	47.2 ± 0.1	58.0 ± 2.2	66.3 ± 1.9	93.9 ± 0.0
Herding [16]	43.9 ± 0.9	44.4 ± 0.6	47.2 ± 0.1	62.7 ± 1.0	71.0 ± 1.6	93.9 ± 0.0
DC-Graph [17]	45.9 ± 0.1	45.8 ± 0.1	47.2 ± 0.1	89.5 ± 0.1	90.5 ± 1.2	93.9 ± 0.0
GCOND [18]	47.1 ± 0.1	47.1 ± 0.1	47.2 ± 0.1	89.6 ± 0.7	90.1 ± 0.5	93.9 ± 0.0
SFGC [19]	47.0 ± 0.1	47.1 ± 0.1	47.2 ± 0.1	90.0 ± 0.3	89.9 ± 0.4	93.9 ± 0.0
GC-SNTK [8]	46.8 ± 0.1	46.5 ± 0.2	47.2 ± 0.1	–	–	–
ICG-NN	50.1 ± 0.2	50.8 ± 0.1	52.7 ± 0.1	89.7 ± 1.3	90.7 ± 1.5	93.6 ± 1.2
IBG-NN	50.5 ± 0.1	51.3 ± 0.2	53.0 ± 0.1	92.3 ± 1.1	92.6 ± 0.6	94.1 ± 0.5

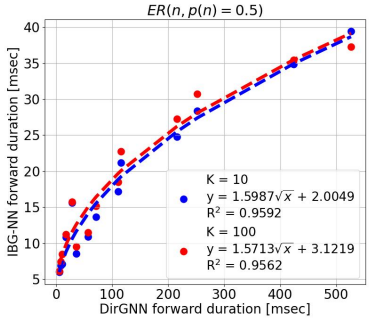


Figure 2: Runtime of K-IBG-NN (K=10, 100) vs. DirGNN on $\mathrm{ER}(n, p = 0.5)$ graphs.

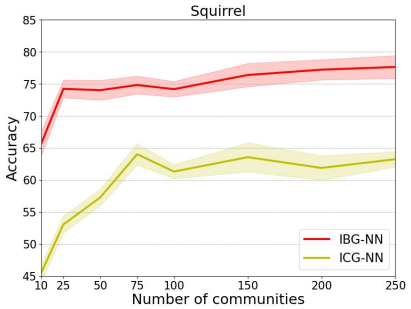


Figure 3: Accuracy of IBG-NN and ICG-NN on Squirrel vs. number of communities.

6.2 Comparing blocks and communities

Setup. We perform a comparison between IBG-NNs and ICG-NNs on Squirrel [21]. We follow the 10 data splits of Pei et al. [21], Li et al. [22] and report the accuracy and standard deviation.

Results. In Figure 3, IBG-NNs exhibit significantly improved performance compared to ICG-NNs. For a small number of communities (10), IBG-NNs achieve 66% accuracy, whereas ICG-NNs achieve only 45%. This performance gap persists as the number of communities increases.

6.3 Subgraph SGD for large graph

Setup. We evaluate IBG-NNs on the large graphs Reddit [1] and Flickr [2], following the data split of [19]. Accuracy and standard deviation are reported over 5 different seeds for varying condensation ratios $r = \frac{M}{N^2}$, where N is the number of nodes, and M is the number of sampled entries of the graph adjacency matrix (See Appendix H for further details). Baseline methods are taken from [19, 8]. We note that a condensation of 100% corresponds to a standard GCN for the baseline methods.

Results. Table 1 demonstrates that subgraph SGD IBG-NN achieves state-of-the-art performance across all sampling rates, surpassing all other coarsening and condensation methods that operate on the full graph in memory, while also improving upon the performance of its predecessor, ICG-NN.

Further ablations, experiments, and a link to our public codebase are provided in Appendix M.

7 Conclusion

We proved a new semi-constructive version of the weak regularity lemma, in which the number of communities needed for a given approximation error is independent of all graph properties, including size and sparsity. In contrast, in previous formulations of the lemma the number of communities increases as the graph becomes sparser. Our formulation is achieved by introducing the densifying cut similarity, which, when optimized, leads the approximating IBG to effectively densify the target graph. This enables fitting IBGs of very low rank ($K = \mathcal{O}(1)$) to large sparse graphs, while previous works required the target graph to be dense for low rank approximations. We introduced IBG-NNs, a network operating on the IBG rather than the original graph, with $\mathcal{O}(N)$ time and memory complexity. IBG-NNs demonstrate state-of-the-art performance in multiple domains: node classification on directed graphs, spatio-temporal graph analysis, and knowledge graph completion.

Acknowledgments

This research was supported by a grant from the United States-Israel Binational Science Foundation (BSF), Jerusalem, Israel, and the United States National Science Foundation (NSF), (NSF-BSF, grant No. 2024660), and by the Israel Science Foundation (ISF grant No. 1937/23).

References

[1] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, 2017.

[2] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. Graphsaint: Graph sampling based inductive learning method. In *ICLR*, 2019.

[3] William L Hamilton, Rex Ying, and Jure Leskovec. Representation learning on graphs: Methods and applications. *IEEE Data Engineering Bulletin*, 2017.

[4] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *ICLR*, 2018.

[5] Stanley Kok and Pedro Domingos. Statistical predicate invention. In *ICML*, 2007.

[6] Emanuele Rossi, Fabrizio Frasca, Ben Chamberlain, Davide Eynard, Michael Bronstein, and Federico Monti. Sign: Scalable inception graph neural networks. In *ICML 2020 Workshop on Graph Representation Learning and Beyond*, 2020.

[7] Wei Jin, Xianfeng Tang, Haoming Jiang, Zheng Li, Danqing Zhang, Jiliang Tang, and Bing Yin. Condensing graphs via one-step gradient matching. In *KDD*, 2022.

[8] Lin Wang, Wenqi Fan, Jiatong Li, Yao Ma, and Qing Li. Fast graph condensation with structure-based neural tangent kernel. In *WWW*, 2024.

[9] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. In *NeurIPS*, 2018.

[10] Filippo Maria Bianchi, Daniele Grattarola, and Cesare Alippi. Spectral clustering with graph neural networks for graph pooling. In *ICML*, 2020.

[11] Ben Finkelshtein, İsmail İlkan Ceylan, Michael Bronstein, and Ron Levie. Learning on large graphs using intersecting communities. In *NeurIPS*, 2024.

[12] Mahdisoltani Farzaneh, Asia Biega Joanna, and M. Suchanek Fabian. Yago3: A knowledge base from multilingual wikipedias. In *CIDR*, 2015.

[13] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *AAAI*, 2018.

[14] Alan Frieze and Ravi Kannan. Quick approximation to matrices and applications. *Combinatorica*, 1999.

[15] Zengfeng Huang, Shengzhong Zhang, Chong Xi, Tang Liu, and Min Zhou. Scaling up graph neural networks via graph coarsening. In *KDD*, 2021.

[16] Max Welling. Herding dynamical weights to learn. In *ICML*, 2009.

[17] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. In *ICLR*, 2020.

[18] Wei Jin, Lingxiao Zhao, Shichang Zhang, Yozen Liu, Jiliang Tang, and Neil Shah. Graph condensation for graph neural networks. In *ICLR*, 2021.

[19] Xin Zheng, Miao Zhang, Chunyang Chen, Quoc Viet Hung Nguyen, Xingquan Zhu, and Shirui Pan. Structure-free graph condensation: From large-scale graphs to condensed graph-free data. In *NeurIPS*, 2024.

[20] Emanuele Rossi, Bertrand Charpentier, Francesco Di Giovanni, Fabrizio Frasca, Stephan Günnemann, and Michael M Bronstein. Edge directionality improves learning on heterophilic graphs. In *LoG*, 2024.

[21] Hongbin Pei, Bingzhe Wei, Kevin Chen-Chuan Chang, Yu Lei, and Bo Yang. Geom-gcn: Geometric graph convolutional networks. In *ICLR*, 2020.

[22] Xiang Li, Renyu Zhu, Yao Cheng, Caihua Shan, Siqiang Luo, Dongsheng Li, and Weining Qian. Finding global homophily in graph neural networks when meeting heterophily. In *ICML*, 2022.

[23] Daniel Zilberg and Ron Levie. Pieclam: A universal graph autoencoder based on overlapping inclusive and exclusive communities. In *ICML*, 2025.

[24] Jaewon Yang and Jure Leskovec. Overlapping community detection at scale: a nonnegative matrix factorization approach. In *Association for Computing Machinery*, 2013.

[25] Thomas Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[26] Simon Geisler, Yujia Li, Daniel J Mankowitz, Ali Taylan Cemgil, Stephan Günnemann, and Cosmin Paduraru. Transformers meet directed graphs. In *ICML*, 2023.

[27] Christian Koke and Daniel Cremers. Holonets: Spectral convolutions do extend to directed graphs. In *ICLR*, 2024.

[28] Ben Finkelshtein, Xingyue Huang, Michael Bronstein, and İsmail İlkan Ceylan. Cooperative graph neural networks. In *ICML*, 2024.

[29] Sohir Maskey, Raffaele Paolino, Aras Bacho, and Gitta Kutyniok. A fractional graph laplacian approach to oversmoothing. In *NeurIPS*, 2024.

[30] Jie Chen, Tengfei Ma, and Cao Xiao. FastGCN: Fast learning with graph convolutional networks via importance sampling. In *ICLR*, 2018.

[31] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph convolutional networks on node classification. In *ICLR*, 2020.

[32] Matthias Fey, Jan-Gin Yuen, and Frank Weichert. Hierarchical inter-message passing for learning on molecular graphs. *arXiv*, 2020.

[33] Houquan Zhou, Shenghua Liu, Danai Koutra, Huawei Shen, and Xueqi Cheng. A provable framework of learning graph embeddings via summarization. In *AAAI*, 2023.

[34] László Miklós Lovász. Large networks and graph limits. In *volume 60 of Colloquium Publications*, 2012.

[35] Ron Levie. A graphon-signal analysis of graph neural networks. In *NeurIPS*, 2023.

[36] Sohir Maskey, Ron Levie, and Gitta Kutyniok. Transferability of graph neural networks: An extended graphon approach. *Applied and Computational Harmonic Analysis*, 2023.

[37] Christian Borgs, Jennifer T Chayes, László Lovász, Vera T Sós, and Katalin Vesztergombi. Convergent sequences of dense graphs i: Subgraph frequencies, metric properties and testing. *Advances in Mathematics*, 2008.

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

[39] Zuoyu Yan, Tengfei Ma, Liangcai Gao, Zhi Tang, and Chao Chen. Link prediction with persistent homology: An interactive view. In *ICML*, 2021.

[40] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *ICLR*, 2019.

[41] Rui Li, Jianan Zhao, Chaozhuo Li, Di He, Yiqi Wang, Yuming Liu, Hao Sun, Senzhang Wang, Weiwei Deng, Yanming Shen, et al. House: Knowledge graph embedding with householder parameterization. In *ICML*, 2022.

[42] Zhaocheng Zhu, Zuobai Zhang, Louis-Pascal Xhonneux, and Jian Tang. Neural bellman-ford networks: A general graph neural network framework for link prediction. In *NeurIPS*, 2021.

[43] Balázs Szegedy László Miklós Lovász. Szemerédi's lemma for the analyst. *GAFA Geometric And Functional Analysis*, 2007.

[44] Yousef Saad. *Numerical methods for large eigenvalue problems: revised edition*. SIAM, 2011.

[45] Lloyd N. Trefethen and David Bau. *Numerical Linear Algebra, Twenty-fifth Anniversary Edition*. Society for Industrial and Applied Mathematics, 2022.

[46] Petros Drineas, Ravi Kannan, and Michael W Mahoney. Fast monte carlo algorithms for matrices i: Approximating matrix multiplication. *SIAM Journal on Computing*, 2006.

[47] Fang Yujie, Li Xin, Ye Rui, Tan Xiaoyan, Zhao Peiyao, and Wang Mingzhong. Relation-aware graph convolutional networks for multi-relational network alignment. *ACM Transactions on Intelligent Systems and Technology*, 2023.

[48] Wang Changping, Wang Chaokun, Wang Zheng, Ye Xiaojun, and S. Yu Philip. Edge2vec. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2020.

[49] Zaheer Manzil, Kottur Satwik, Ravanbakhsh Siamak, Póczos Barnabás, Salakhutdinov Ruslan, and Smola Alex. Deep sets. In *NeurIPS*, 2017.

[50] Oleg Platonov, Denis Kuznedelev, Michael Diskin, Artem Babenko, and Liudmila Prokhorenkova. A critical look at the evaluation of GNNs under heterophily: Are we really making progress? In *ICLR*, 2023.

[51] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018.

[52] Jiong Zhu, Yujun Yan, Lingxiao Zhao, Mark Heimann, Leman Akoglu, and Danai Koutra. Beyond homophily in graph neural networks: Current limitations and effective designs. In *NeurIPS*, 2020.

[53] Eli Chien, Jianhao Peng, Pan Li, and Olgica Milenkovic. Adaptive universal generalized pagerank graph neural network. In *ICLR*, 2020.

[54] Derek Lim, Felix Hohne, Xiuyu Li, Sijia Linda Huang, Vaishnavi Gupta, Omkar Bhalerao, and Ser Nam Lim. Large scale learning on non-homophilous graphs: New benchmarks and strong simple methods. In *NeurIPS*, 2021.

[55] Sunil Kumar Maurya, Xin Liu, and Tsuyoshi Murata. Improving graph neural networks with simple architecture design. *arXiv*, 2021.

[56] Sitao Luan, Chenqing Hua, Qincheng Lu, Jiaqi Zhu, Mingde Zhao, Shuyuan Zhang, Xiao-Wen Chang, and Doina Precup. Is heterophily a real nightmare for graph neural networks to do node classification? *arXiv*, 2021.

[57] T Konstantin Rusch, Benjamin P Chamberlain, Michael W Mahoney, Michael M Bronstein, and Siddhartha Mishra. Gradient gating for deep multi-rate learning on graphs. In *ICLR*, 2022.

[58] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. Graph wavenet for deep spatial-temporal graph modeling. In *IJCAI*, 2019.

[59] Lei Bai, Lina Yao, Can Li, Xianzhi Wang, and Can Wang. Adaptive graph convolutional recurrent network for traffic forecasting. In *NeurIPS*, 2020.

[60] Andrea Cini, Ivan Marisca, Daniele Zambon, and Cesare Alippi. Taming local effects in graph-based spatiotemporal forecasting. In *NeurIPS*, 2024.

[61] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *NeurIPS*, 2013.

[62] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR*, 2014.

[63] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *ICML*, 2016.

[64] Fan Yang, Zhilin Yang, and William W Cohen. Differentiable learning of logical rules for knowledge base reasoning. In *NeurIPS*, 2017.

[65] Meng Qu, Junkun Chen, Louis-Pascal Xhonneux, Yoshua Bengio, and Jian Tang. Rnnlogic: Learning logic rules for reasoning on knowledge graphs. In *ICLR*, 2020.

[66] Kewei Cheng, Jiahao Liu, Wei Wang, and Yizhou Sun. Rlogic: Recursive logical rule learning from knowledge graphs. In *KDD*, 2022.

[67] Kewei Cheng, Nesreen K Ahmed, and Yizhou Sun. Neural compositional rule learning for knowledge graph reasoning. In *ICLR*, 2023.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We prove our version of the weak regularity lemma, showing how our proposed factorization allows approximating any graph by reweighting the contribution of non-edges. We propose a neural network operating on the factorization and empirically show it achieves strong performance with high efficiency.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of our method in the appendix, stating that our approach may fall short in domains where small scale local structures are crucial. Furthermore, we explain our method is only applicable for a single graph at a time, and cannot be used for graph-level tasks.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: In the Appendix, we provide rigorous proofs for all theoretical results in our paper. We provide all relevant assumptions in a coherent and accessible manner.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide all the relevant information for recreating all results presented in the paper, including model architecture, hyperparameters searched, datasets tested, hardware used and any other relevant material.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We provide a URL to the code and commands for reproducing the different experiments in the main paper. We further provide the hyperparameter grids for all experiments in the appendix.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: In the experiments section and the appendix, we provide all data splits and full experimental settings, including hyperparameters, optimizers, hardware, and any other necessary information. We follow standard, commonly used splits.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report the standard deviation for all experiments, figures, and tables.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We report the hardware used in the Appendix. We use a single NVIDIA L40 GPU for all experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research fully complies with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: We propose a method for the approximation of general graphs and a procedure for utilizing the approximation for efficient graph learning. Our work has a strong focus on theoretical results, hence, we believe it has no meaningful societal impact.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: Our work poses no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We cite the relevant data sources and credit them for every dataset used.

Guidelines:

- The answer NA means that the paper does not use existing assets.

- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: We do not release any new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Related work

Intersecting Community Graphs (ICG). Our work continues the ICG setting of [11], which introduces a weak regularity lemmas for practical graph computations. The work of [11] presented a pipeline for operating on undirected, non-sparse graphs. Similarly to our work, [11] follow a two stage procedure. In the first stage, the graph is approximated by learning a factorization into undirected communities, forming what they call an Intersecting Community Graph (ICG). In the second stage, the ICG is used to enrich a neural network operating on the node features and community graph without using the original edge connectivity. This setup allows for more efficient computation, both in terms of runtime and memory, since the full edge structure of the graph, used in standard GNNs, can be replaced with a much smaller community-level graph—especially useful for graphs with a high average degree. The constructive weak regularity lemma presented in [11] shows any graph can be approximated in *cut-norm*, regardless of its size, by minimizing the easy to compute Frobenius error.

More concretely, an ICG with K communities is just like an IBG with only node features (no edge-features), but with the transmitting and receiving communities being equal $\mathbf{U} = \mathbf{V}$. Namely, an ICG can be represented by a triplet of *community affiliation matrix* $\boldsymbol{Q} \in \mathbb{R}^{N\times K}$, *community magnitude vector* $\boldsymbol{r} \in \mathbb{R}^{K}$, and *community feature matrix* $\mathbf{F} \in \mathbb{R}^{K\times D}$. An ICG $(\boldsymbol{C}, \boldsymbol{P})$ with adjacency matrix $\boldsymbol{C}$ and signal $\boldsymbol{P}$ is then given by

$$\boldsymbol{C} = \boldsymbol{Q}\,\mathrm{diag}(\boldsymbol{r})\boldsymbol{Q}^{\top} \quad \text{and} \quad \boldsymbol{P} = \boldsymbol{Q}\boldsymbol{F},$$

where $\mathrm{diag}(\boldsymbol{r})$ is the diagonal matrix in $\mathbb{R}^{K\times K}$ with $\boldsymbol{r}$ as its diagonal elements. Here, K is the number of communities, N is the number of nodes, and E is the number of edges.

When approximating a graph-signal $(\boldsymbol{A}, \boldsymbol{X})$, the measure of accuracy, or error, in [11] is defined to be the standard (unweighted) cut metric $\|(\boldsymbol{A}, \boldsymbol{X}) - (\boldsymbol{C}, \boldsymbol{P})\|_{\square}$. The semi-constructive regularity lemma of [11] states that it is enough to minimize the standard Forbenius error $\|(\boldsymbol{A}, \boldsymbol{X}) - (\boldsymbol{C}, \boldsymbol{P})\|_{\mathrm{F}}$ in order to guarantee

$$\|(\boldsymbol{A}, \boldsymbol{X}) - (\boldsymbol{C}, \boldsymbol{P})\|_{\square} = \mathcal{O}(N/\sqrt{KE}), \tag{9}$$

Looking at (9) it is clear that in order to guarantee a small approximation error in cut metric, the number of communities must increase as $N/\sqrt{E}$ becomes larger. Specifically, the number of communities K is independent of the size of the graph only when $E = N^2$, i.e., the graph is dense. Hence, the ICG method falls short for sparse graphs

Our IBG method solves this shortcoming and more. For example, a main contribution of our method is a densification mechanism, supported by our novel densifying cut similarity measure and our densifying regularity lemma, which is a non-trivial continuation and extension of the semi-constructive weak regularity lemma of [11]. Please see Appendix J for a detailed comparison of our IBG method to ICG.

Cluster Affiliation models (BigClam and PieClam). A similar work is PieClam [23], extending the well known BigClam model [24], which builds a probabilistic model of graphs as intersections of overlapping communities. While BigClam only allows communities with positive coefficients, which limits the ability to approximate many graphs, like bipartite graphs, PieClam formulates a graph probabilistic autoencoder that also includes negative communities. This allows approximately encoding any dense graph with a fixed budget of parameters per node. We note that as opposed to ICG and PieClam, which can only theoretically approximate dense symmetric graphs with $\mathcal{O}(1)$ communities, our IBG method can approximate both sparse and dense (non-symmetric in general) graphs with $\mathcal{O}(1)$ communities via the densification mechanism (the densifying constructive regularity lemma with respect to the densifying cut similarity).

GNNs for directed graphs. The standard practice in GNN design is to assume that the graph is undirected [25]. However, this assumption not only alters the input data by discarding valuable directional information, but also overlooks the empirical evidence demonstrating that leveraging edge directionality can significantly enhance performance [20]. For instance, DirGNN [20] extends message-passing neural networks (MPNNs) to directed graphs, while [26] adapts transformers for the same purpose. FaberNet [27] generalizes spectral convolutions to directed graphs, all of which have led to improved performance. Co-GNN [28] demonstrates the advantage of learning edge directionality over using conventional undirected graph representations. Furthermore, the proper handling of directed edges has enabled [29] to extend the concept of oversmoothing to directed

graphs, providing deeper theoretical insights. IBG-NNs also capitalize on edge directionality, achieving notable performance improvements over their predecessor ICG-NNs [11], as demonstrated in Appendix M.2.1. When compared to existing GNNs designed for directed graphs, IBG-NNs offer a more efficient approach to signal processing. Specifically, for IBG-NNs to outperform message-passing-based GNNs in terms of efficiency, the condition $KN < E$ must hold. Such a choice of K typically produces good performance for most graphs. This efficiency advantage allows IBG-NNs to make better use of the input edges while being more efficient than traditional GNNs.

Graph Pooling GNNs. Graph pooling GNNs generate a sequence of increasingly coarsened graphs by aggregating neighboring nodes into "super-nodes" [9, 10], where standard message-passing is applied on the intermediate coarsened graphs. Similarly, in IBG-NNs, the signal is projected, but onto overlapping blocks rather than disjoint clusters, with several additional key distinctions: (1) The blocks in IBG-NNs are overlapping and cover large regions of the graph, allowing the method to preserve fine-grained, high-frequency signal details during projection, unlike traditional graph pooling approaches. (2) Operations on community features in IBG-NNs possess a global receptive field, enabling the capture of broader structural patterns across the graph – an extremely difficult task for local graph pooling approach. (3) IBG-NNs diverge from the conventional message-passing framework: the flattened community feature vector, which lacks symmetry, is processed by a general multilayer perceptron (MLP), whereas message-passing neural networks (MPNNs) apply the same function uniformly to all edges. (4) IBG-NNs operate exclusively on an efficient data structure, offering both theoretical guarantees and empirical evidence of significantly improved computational efficiency compared to graph pooling methods.

Graph reduction methods. Graph reduction aims to reduce the size of the graph while preserving key information. It can be categorized into three main approaches: graph sparsification, graph coarsening and graph condensation. Graph sparsification methods [1, 2, 30, 31] approximate a graph by retaining only a subset of its edges and nodes, often employing random sampling techniques. Graph coarsening [32, 15] clusters sets of nodes into super-nodes while aggregating inter-group edges into super-edges, aiming to preserve structural properties such as the degree distribution [33]. Graph condensation [18] generates a smaller graph with newly created nodes and edges, designed to maintain the performance of GNNs on downstream tasks.

While subgraph SGD in IBG-NNs also involves subsampling, it differs fundamentally by providing a provable approximation of the original graph. This contrasts with graph sparsification for example – where some, hopefully good heuristic-based sampling is often employed. More importantly, subgraph IBG-NNs offer a subgraph sampling approach for cases where the original graph is too large to fit in memory. This contrasts with the aforementioned coarsening and condensation methods, which lack a strategy for managing smaller data structures during the computation of the compressed graph.

Graph reduction methods generally rely on locality, applying message-passing on the reduced graph. In particular, condensation techniques require $\mathcal{O}(EM)$ operations to construct a smaller graph [18, 19, 8], where E is the number of edges in the original graph and M is the number of nodes in the condensed graph. In contrast, IBG-NNs estimate the IBG with only $\mathcal{O}(E)$ operations.

Furthermore, while conventional reduction methods process representations on either an iteratively coarsened graph or mappings between the full and reduced graphs, IBG-NNs incorporate fine-grained node information at every layer, leading to richer representations.

Cut metric in graph machine learning. The cut metric is a useful similarity measure, which can separate any non-isomorphic graphons [34]. This makes the cut metric particularity useful in deriving new theoretical insights for graph machine learning. For instance, [35] demonstrated that GNNs with normalized sum aggregation cannot separate graph-signals that are close to each other in cut metric. Using the cut distance as a theoretical tool, [36] proves that spectral GNNs with continuous filters are transferable between graphs in sequences of that converge in homomorphism density. [11] introduced a semi constructive weak regularity lemma and used it to build new algorithms on large undirected non-sparse graphs. In this work we introduce a new graph similarity measure – the densifying cut similarity, which gives higher importance to edges than non-edges in a graph. This allows us to approximate any graph using a set of overlapping bipartite components, where the size of the set only depends on the error tolerance. Similarly to [11], we present a semi constructive weak regularity lemma. As oppose to [11], using our novel similarity measure, our regularity lemma can be used to build new algorithms on large directed graphs which are sparse.

B The weak regularity lemma

Consider a graph G with a node set $\mathcal{V} = [N] = \{1, \ldots, N\}$ and an edge set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. We define an equipartition $\mathcal{P} = \{\mathcal{V}_1, \ldots \mathcal{V}_k\}$ as a partition of $\mathcal{V}$ into k sets where $||\mathcal{V}_i| - |\mathcal{V}_j|| \leq 1$ for every $1 \leq i, j \leq k$. For any pair of subsets $\mathcal{U}, \mathcal{S} \subset \mathcal{V}$ denote by $e_G(\mathcal{U}, \mathcal{S})$ the number of edges between $\mathcal{U}$ and $\mathcal{S}$. Now, consider two node subsets $\mathcal{U}, \mathcal{S} \subset \mathcal{V}$. If the edges between $\mathcal{V}_i$ and $\mathcal{V}_j$ were to be uniformly and independently distributed, then the expected number of edges between $\mathcal{U}$ and $\mathcal{S}$ would be

$$e_{\mathcal{P}(\mathcal{U},\mathcal{S})} \coloneqq \sum_{i=1}^{k} \sum_{j=1}^{k} \frac{e_G(\mathcal{V}_i, \mathcal{V}_j)}{|\mathcal{V}_i|\,|\mathcal{V}_j|} \,|\mathcal{V}_i \cap \mathcal{U}|\,|\mathcal{V}_j \cap \mathcal{S}|\,.$$

Using the above, we define the *irregularity*:

$$\mathrm{irreg}_G(\mathcal{P}) = \max_{\mathcal{U},\mathcal{S} \subset \mathcal{V}} |e_G(\mathcal{U}, \mathcal{S}) - e_{\mathcal{P}}(\mathcal{U}, \mathcal{S})| \,/\, |\mathcal{V}|^2\,. \tag{10}$$

The *irregularity* measures how non-random like the edges between $\{\mathcal{V}_j\}$ behave.

We now present the weak regularity lemma.

Theorem B.1 (Weak Regularity Lemma [14])**.** *For every $\epsilon > 0$ and every graph $G = (\mathcal{V}, \mathcal{E})$, there is an equipartition $\mathcal{P} = \{\mathcal{V}_1, \ldots, \mathcal{V}_k\}$ of $\mathcal{V}$ into $k \leq 2^{c/\epsilon^2}$ classes such that* $\mathrm{irreg}_G(\mathcal{P}) \leq \epsilon$*. Here, c is a universal constant that does not depend on G and ϵ.*

The weak regularity lemma states that any large graph G can be approximated by a weighted graph G^ϵ with node set $\mathcal{V}^\epsilon = \{\mathcal{V}_1, \ldots, \mathcal{V}_k\}$. The nodes of G^ϵ represent clusters of nodes from G, and the edge weight between two clusters $\mathcal{V}_i$ and $\mathcal{V}_j$ is given by $\frac{e_G(\mathcal{V}_i, \mathcal{V}_j)}{|\mathcal{V}_i||\mathcal{V}_j|}$. In this context, an important property of the irregularity $\mathrm{irreg}_G(\mathcal{P})$ is that it can be seen as the cut metric between the G and a SBM based on G^ϵ. For each node i denote by $\mathcal{V}_{q_i} \in \mathcal{P}$ the partition set that contains the node. Given G^ϵ, construct a new graph $G^{\mathcal{P}}$ with N nodes, whose adjacency matrix $\boldsymbol{A}^{\mathcal{P}} = (a^{\mathcal{P}}_{i,j})_{i,j=1}^{|\mathcal{V}|}$ is defined by

$$a^{\mathcal{P}}_{i,j} = \frac{e_G(\mathcal{V}_{q_i}, \mathcal{V}_{q_j})}{|\mathcal{V}_{q_i}||\mathcal{V}_{q_j}|},$$

Let $\boldsymbol{A}$ be the adjacency matrix of G. It can be shown that

$$\left\|\boldsymbol{A} - \boldsymbol{A}^{\mathcal{P}}\right\|_{\square} = \mathrm{irreg}_G(\mathcal{P}),$$

which shows that the weak regularity lemma can be expressed in terms of cut norm rather than irregularity.

C Graphons and norms

Kernel. A kernel Y is a measurable function $Y : [0,1]^2 \to [-1,1]$.

Graphon. A graphon [37, 34] is a measurable function $W : [0,1]^2 \to [0,1]$. A graphon can be seen as a weighted graph, where the node set is the interval $[0,1]$, and for any pair of nodes $x, y \in [0,1]$, the weight of the edge between x and y is $W(x,y)$, which can also be seen as the probability of having an edge between x and y. We note that in the standard definition a graphon is defined to be symmetric, but we remove this restriction in our construction.

Kernel-signal and Graphon-signal. A kernel-signal is a pair (Y, y) where Y is a kernel and $y : [0,1] \to \mathbb{R}^D$ is a measurable function. A graphon-signal is defined similarly with a graphon in place of a kernel.

Induced graphon-signal. Consider an interval equipartition $\mathcal{I}_m = \{I_1, \ldots, I_m\}$, a partition of $[0,1]$ into disjoint intervals of equal length. Given a graph G with an adjacency matrix $\boldsymbol{A}$, the induces graphon $W_{\boldsymbol{A}}$ is the graphon defined by $W_{\boldsymbol{A}}(x,y) = \boldsymbol{A}_{\lceil xm \rceil \lceil ym \rceil}$, where we use the convention that $\lceil 0 \rceil = 1$. Notice that $W_{\boldsymbol{A}}$ is a piecewise constant function on $\mathcal{I}_m \times \mathcal{I}_m$. As such, a graph of m nodes can be identified by its induced graphon that is piecewise constant on $\mathcal{I}_m \times \mathcal{I}_m$.

C.1 Weighted Frobenius and cut norm

Weighted Frobenius norm. Let $q : [0,1]^2 \to [c,\infty)$ be a measurable function in $\mathcal{L}^\infty([0,1]^2)$, where $c > 0$. We call such a q a *weight function*. Consider the real weighted Lebesgue space $\mathcal{L}^2([0,1]^2; q)$ defined with the inner product

$$\langle Y, Y' \rangle_q := \frac{1}{\|1\|_{1;q}} \iint_{[0,1]^2} Y(x,y)Y'(x,y)q(x,y)dxdy,$$

where 1 is the constant function $[0,1]^2 \ni (x,y) \mapsto 1$ and $\|1\|_{1;q} = \iint q$. When $q = 1$, we denote $\langle X, Z \rangle := \langle X, Z \rangle_1$. Let $\alpha, \beta > 0$. Consider the real Hilbert space $L^2([0,1]^2; q) \times (L^2[0,1])^D$ defined with the weighted inner product

$$\begin{aligned}&\langle (Y,y), (Y',y') \rangle_q = \langle (Y,y), (Y',y') \rangle_{q,\alpha,\beta} \\ &= \alpha \frac{1}{\|1\|_{1;q}} \iint_{[0,1]^2} Y(x,y)Y'(x,y)q(x,y)dxdy + \frac{\beta}{D} \sum_{j=1}^{D} \int_{[0,1]} y_j(x)y'_j(x)dx.\end{aligned}$$

We call the corresponding weighted norm the *weighted Frobenius norm*, denoted by

$$\|(Y,y)\|_{\mathrm{F};q} = \|(Y,y)\|_{\mathrm{F};q,\alpha,\beta} = \sqrt{\alpha \|Y\|^2_{\mathrm{F};q} + \frac{\beta}{D} \sum_{j=1}^{D} \|y_j\|^2_{\mathrm{F}}},$$

where $\|Y\|^2_{\mathrm{F};q} = \langle Y, Y \rangle_q$ and $\|y_j\|^2_{\mathrm{F}} = \langle y_j, y_j \rangle_1$.

Similarly, for a matrix-signal, we consider a *weight matrix* $\boldsymbol{Q} \in [c,\infty)^{N \times N}$, where $c > 0$. For $\boldsymbol{D}, \boldsymbol{D}' \in \mathbb{R}^{N \times N}$, define the weighted inner product by

$$\langle \boldsymbol{D}, \boldsymbol{D}' \rangle_{\boldsymbol{Q}} := \frac{1}{\|\mathbf{1}\|_{1;\boldsymbol{Q}}} \sum_{i,j \in [N]^2} d_{i,j} d'_{i,j} q_{i,j},$$

where $\mathbf{1} \in \mathbb{R}^{N \times N}$ is the matrix with all entries equal to 1, and $\|\mathbf{1}\|_{1;\boldsymbol{Q}} = \sum_{i,j \in [N]^2} q_{i,j}$. Define the *weighted matrix-signal Frobenius norm* by

$$\|(\boldsymbol{D}, \boldsymbol{Z})\|_{\mathrm{F};\boldsymbol{Q}} = \|(\boldsymbol{D}, \boldsymbol{Z})\|_{\mathrm{F};\boldsymbol{Q},\alpha,\beta} = \sqrt{\alpha \|\boldsymbol{D}\|^2_{\mathrm{F};\boldsymbol{Q}} + \frac{\beta}{D} \sum_{j=1}^{D} \|\boldsymbol{z}_j\|^2_{\mathrm{F}}},$$

where $\|\boldsymbol{D}\|^2_{\mathrm{F};\boldsymbol{Q}} = \langle \boldsymbol{D}, \boldsymbol{D} \rangle_{\boldsymbol{Q}}$ and $\|\boldsymbol{z}_j\|^2_{\mathrm{F}} = \langle \boldsymbol{z}_j, \boldsymbol{z}_j \rangle_1$.

Graphon weighted cut norm and cut metric. Define for a kernel-signal (Y, y) the *weighted cut norm*

$$\|(Y,y)\|_{\square;q,\alpha,\beta} = \|(Y,y)\|_{\square;q} = \frac{\alpha}{\|1\|_{1;q}} \sup_{\mathcal{U},\mathcal{V}} \left| \int_{\mathcal{U}} \int_{\mathcal{V}} Y(x,y)q(x,y)dxdy \right| + \beta \frac{1}{D} \sum_{j=1}^{D} \sup_{\mathcal{U}} \left| \int_{\mathcal{U}} y_j(x)dx \right|,$$

where the supremum is over the set of measurable subsets $\mathcal{U}, \mathcal{V} \subset [0,1]$.

The weighted cut metric between two graphon-signals (W, f) and (W', f') is defined to be $\|(W,f) - (W',f')\|_{\square;q}$.

Graph-signal weighted cut norm and cut metric. Define for a matrix-signal $(\boldsymbol{D}, \boldsymbol{Z})$, where $\boldsymbol{D} \in [-1,1]^{N \times N}$ and $\boldsymbol{Z} \in \mathbb{R}^{N \times D}$, the *weighted cut norm*

$$\begin{aligned}&\|(\boldsymbol{D}, \boldsymbol{Z})\|_{\square;\boldsymbol{Q},\alpha,\beta} = \|(\boldsymbol{D}, \boldsymbol{Z})\|_{\square;\boldsymbol{Q}} = \alpha \|\boldsymbol{D}\|_{\square;\boldsymbol{Q}} + \beta \|\boldsymbol{Z}\|_{\square} \\ &= \frac{\alpha}{\|\mathbf{1}\|_{1;\boldsymbol{Q}}} \max_{\mathcal{U},\mathcal{V} \subset [N]} \left| \sum_{i \in \mathcal{U}} \sum_{j \in \mathcal{V}} d_{i,j} q_{i,j} \right| + \frac{\beta}{DN} \sum_{j=1}^{D} \max_{\mathcal{U} \subset [N]} \left| \sum_{i \in \mathcal{U}} z_{i,j} \right|.\end{aligned}$$

The weighted cut metric between two graph-signals $(\boldsymbol{A}, \boldsymbol{s})$ and $(\boldsymbol{A}', \boldsymbol{s}')$ is defined to be $\|(\boldsymbol{A}, \boldsymbol{s}) - (\boldsymbol{A}', \boldsymbol{s}')\|_{\square;\boldsymbol{Q}}$. We note that this metric gives a meaningful notion of graph-signal similarity for graphs as long as their number of edges satisfy $\|\mathbf{1}\|_{1;\boldsymbol{Q}} = \Theta(E)$[2]. All graphs with $E \ll \|\mathbf{1}\|_{1;\boldsymbol{Q}}$ have distance close to zero from each other, so the cut metric does not have a meaningful or useful separation power for such graphs.

Densifying cut similarity. In this paper, we will focus on a special construction of a weighted cut norm, which we construct and motivate next.

In graph completion tasks, such as link prediction or knowledge graph reasoning, the objective is to complete a partially observed adjacency matrix. Namely, there is a set of known dyads[3] $\mathcal{M} \subset [N]^2$, and the given data is the restriction of $\boldsymbol{A}$ to the known dyads

$$\boldsymbol{A}|_{\mathcal{M}} : \mathcal{M} \to \{0, 1\}.$$

The goal is then to find an adjacency matrix $\boldsymbol{B}$ that fits $\boldsymbol{A}$ on the known dyads, namely, $\boldsymbol{B}|_{\mathcal{M}} \approx \boldsymbol{A}|_{\mathcal{M}}$, with the hope that $\boldsymbol{B}$ also approximates $\boldsymbol{A}$ on the unknown dyads due to some inductive bias.

Recall that $\mathcal{E}$ denotes the set of edges of $\boldsymbol{A}$. We call $\mathcal{E}^{\mathrm{c}} = [N]^2 \setminus \mathcal{E}$ the set of *non-edges*. The training set in graph completion consists of the edges $\mathcal{E} \cap \mathcal{M}$ and the non-edges $\mathcal{E}^{\mathrm{c}} \cap \mathcal{M}$. Typical methods, such as VGAE [38] and TLC-GNN [39], define a loss of the form

$$l(\boldsymbol{B}) = \sum_{(n,m)\in\mathcal{M}\cap\mathcal{E}} c_{n,m}\psi_1(b_{n,m}, a_{n,m}) + \sum_{(n,m)\in\mathcal{M}\cap\mathcal{E}^{\mathrm{c}}} c_{n,m}\psi_2(b_{n,m}, a_{n,m}),$$

where $\psi_1, \psi_2 : \mathbb{R}^2 \to \mathbb{R}_+$ are dyad-wise loss functions and $c_{n,m} \in \mathbb{R}_+$ are weights. Many methods, like RotateE [40], HousE [41] and NBFNet [42], give one weight $c_{n,m} = C$ for edges $(n, m) \in \mathcal{E}\cap\mathcal{M}$ and a smaller weight $c_{n,m} = c \ll C$ for non-edges $(n, m) \in \mathcal{E}^{\mathrm{c}} \cap \mathcal{M}$. The motivation is that for sparse graphs there are many more non-edges than edges, and giving the edges and non-edges the same weight would tend to produce learned $\boldsymbol{B}$ that does not put enough emphasis on the connectivity structure of $\boldsymbol{A}$. In practice, the smaller weight for non-edges is implemented implicitly by taking random samples from $\mathcal{M}$ during training, balancing the number of samples from $\mathcal{E} \cap \mathcal{M}$ and from $\mathcal{E}^{\mathrm{c}} \cap \mathcal{M}$. The samples from $\mathcal{E}^{\mathrm{c}} \cap \mathcal{M}$ are called *negative samples*.

Remark C.1. *In this paper we interpret such an approach as learning a* densified version *of $\boldsymbol{A}$. Namely, by putting less emphasis on non-edges, the matrix $\boldsymbol{B}$ roughly fits the structure of $\boldsymbol{A}$, but with a higher average degree.*

Motivated by the above discussion, we also define a densifying version of cut distance. Given a target unweighted adjacency matrix $\boldsymbol{A} = (a_{i,j})_{i,j=1}^N$ to be approximated, we consider the weight matrix $\boldsymbol{Q} = e\mathbf{1} + (1 - e)\boldsymbol{A}$ for some small e and $\mathbf{1}$ being the all 1 matrix. Denote the number of edges by $E = |\mathcal{E}|$. Next, we would like to choose e to reflect some desired balance between edges and non-edges. Since the number of non-edges is $N^2 - E$ and the number of edges is E, we choose e in such a way that $e - (E/N^2)e = \Gamma E/N^2$ for some $\Gamma > 0$, namely,

$$e = e_{E,\Gamma} = \frac{\Gamma E/N^2}{1 - (E/N^2)}. \tag{11}$$

The interpretation of Γ is the proposition of sampled non-edges when compared with the edges. Namely, the weight matrix $\boldsymbol{Q} = e + (1 - e)\boldsymbol{A}$ effectively simulates taking ΓE negative samples and E samples. Observe that

$$\|\boldsymbol{A}\|^2_{\mathrm{F};e\mathbf{1}+(1-e)\boldsymbol{A}} = \frac{1}{\sum_{i,j\in[N]^2} q_{i,j}} \sum_{i,j\in[N]^2} a_{i,j}q_{i,j} = \frac{E/N^2}{e + (1 - e)E/N^2} = 1/(\Gamma + 1).$$

To standardize the above similarity measure, we normalize it and define the weighted Frobenius norm $(1 + \Gamma)\|\boldsymbol{B}\|_{\mathrm{F};\boldsymbol{Q}_{\boldsymbol{A}}}$ and weighted cut norm $(1 + \Gamma)\|\boldsymbol{B}\|_{\square;\boldsymbol{Q}_{\boldsymbol{A}}}$, where

$$\boldsymbol{Q}_{\boldsymbol{A}} = \boldsymbol{Q}_{\boldsymbol{A},\Gamma} := e_{E,\Gamma}\mathbf{1} + (1 - e_{E,\Gamma})\boldsymbol{A}, \tag{12}$$

[2]The asymptotic notation $a_n = \Theta(b_n)$ means that there exist positive constants c_1, c_2 and n_0 such that $c_1 b_n \le a_n \le c_2 b_n$ for all $n \ge n_0$. In our analysis, we suppose that there is a sequence of graphs with N_n nodes, E_n edges, and weight matrices $\boldsymbol{Q}_n$.

[3]A dyad is a pair of nodes $(m, n) \in [N]^2$. For a simple graph, a dyad may be an edge or a non-edge.

and where $e_{E,\Gamma}$ is defined in (11). We now have

$$(1+\Gamma)\left\|\boldsymbol{A}\right\|_{\mathrm{F};\boldsymbol{Q}_{\boldsymbol{A}}}=1. \tag{13}$$

This standardization assures that merely increasing Γ in the definition of the cut metric $(1+\Gamma)\left\|\boldsymbol{A}-\boldsymbol{B}\right\|_{\mathrm{F};\boldsymbol{Q}_{\boldsymbol{A}}}$ would not lead to a seemingly better approximation. The above discussion leads to the following definition.

Definition C.1. *Let $\boldsymbol{A}\in\{0,1\}^{N\times N}$ be an unweighted adjacency matrix, and $\Gamma>0$. The* densifying cut similarity *between the target $\boldsymbol{A}$ and any adjacency matrix $\boldsymbol{B}\in\mathbb{R}^{N\times N}$ is defined to be*

$$\sigma_{\square}(\boldsymbol{A}||\boldsymbol{B})=\sigma_{\square;\Gamma}(\boldsymbol{A}||\boldsymbol{B}):=(1+\Gamma)\left\|\boldsymbol{A}-\boldsymbol{B}\right\|_{\square;\boldsymbol{Q}_{\boldsymbol{A}}},$$

where $\boldsymbol{Q}_{\boldsymbol{A}}$ is defined in (12). Given $\alpha,\beta>0$ such that $\alpha+\beta=1$, the densifying cut similarity *between the target graph-signal $(\boldsymbol{A},\boldsymbol{X})$ and the graph-signal $(\boldsymbol{A}',\boldsymbol{X}')$ is defined to be*

$$\sigma_{\square}\big((\boldsymbol{A},\boldsymbol{X})||(\boldsymbol{A}',\boldsymbol{X}')\big)=\sigma_{\square;\alpha,\beta,\Gamma}\big((\boldsymbol{A},\boldsymbol{X})||(\boldsymbol{A}',\boldsymbol{X}')\big):=\alpha\sigma_{\square;\Gamma}(\boldsymbol{A}||\boldsymbol{B})+\beta\left\|\boldsymbol{X}-\boldsymbol{X}'\right\|_{\square}.$$

We moreover note that the similarity measure $\sigma_{\square}(\boldsymbol{A}||\boldsymbol{B})$ is not symmetric, and hence not a metric. The first entry $\boldsymbol{A}$ in $\sigma_{\square}(\boldsymbol{A}||\boldsymbol{B})$ is interpreted as the thing to be approximated, and the second entry $\boldsymbol{B}$ as the approximant. Here, when fitting an IBG to a graph, $\boldsymbol{A}$ is a constant, and $\boldsymbol{B}$ is the variable.

D Proof of the semi-constructive densifying directional soft weak regularity lemma

In this section we prove a version of the constructive weak regularity lemma for asymmetric graphon signals. Prior information regarding cut-distance, the original formulation of the weak regularity lemma and it's constructive version for symmetric graphon-signals can be found in (11, Appendix A, B).

D.1 Intersecting block graphons

Below, we extend the definition of IBGs for graphons. The construction is similar to the one in Appendix B.3 of Finkelshtein et al. [11], where ICGs are extended to graphons. Denote by χ the set of all indicator functions of measurable subset of $[0,1]$

$$\chi=\{\mathbb{1}_u\mid u\subset[0,1]\text{ measurable}\}.$$

Definition D.1. *A set $\mathcal{Q}$ of bounded measurable functions $q:[0,1]\to\mathbb{R}$ that contains χ is called a* soft affiliation model*.*

For the case of node level graphon-signals, we use the following definition:

Definition D.2. *Let $D\in\mathbb{N}$. Given a soft affiliation model $\mathcal{Q}$, the subset $[\mathcal{Q}]$ of $L^2[0,1]^2\times(L^2[0,1])^D$ of all elements of the form $(au(x)v(y),bu(z)+cv(z))$, with $u,v\in\mathcal{Q}$, $a\in\mathbb{R}$ and $b,c\in\mathbb{R}^D$, is called the* soft rank-1 intersecting block graphon (IBG) model *corresponding to $\mathcal{Q}$. Given $K\in\mathbb{N}$, the subset $[\mathcal{Q}]_K$ of $L^2[0,1]^2\times(L^2[0,1])^D$ of all linear combinations of K elements of $[\mathcal{Q}]$ is called the* soft rank-K IBG model *corresponding to $\mathcal{Q}$. Namely, $(C,p)\in[\mathcal{Q}]_K$ if and only if it has the form*

$$C(x,y)=\sum_{k=1}^{K}a_ku_k(x)v_k(y)\quad\textit{and}\quad p(z)=\sum_{k=1}^{K}b_ku_k(z)+c_kv_k(z)$$

where $(u_k)_{k=1}^K\in\mathcal{Q}^K$ are called the target community affiliation functions*, $(v_k)_{k=1}^K\in\mathcal{Q}^K$ are called the* source community affiliation functions*, $(a_k)_{k=1}^K\in\mathbb{R}^K$ are called the* community affiliation magnitudes*, $(b_k)_{k=1}^K\in\mathbb{R}^{K\times D}$ are called the* target community features*, and $(c_k)_{k=1}^K\in\mathbb{R}^{K\times D}$ the* source community features*. Any element of $[\mathcal{Q}]_K$ is called an intersecting block graphon-signal (*IBG*).*

D.2 The semi-constructive weak regularity lemma in Hilbert space

In this subsection we prove the constructive weak graphon-signal regularity lemma.

László Miklós Lovász [43] extended the weak regularity lemma to graphons. They showed that the lemma follows from a more general result about approximation in Hilbert spaces – the weak regularity lemma in Hilbert spaces [43, Lemma 4]. We extend this result to have a constructive form, which we later use to prove Theorem 4.1. For completeness, we begin by stating the original weak regularity lemma in Hilbert spaces from [43].

Lemma D.1 ([43])**.** *Let $\mathcal{K}_1, \mathcal{K}_2, \ldots$ be arbitrary nonempty subsets (not necessarily subspaces) of a real Hilbert space $\mathcal{H}$. Then, for every $\epsilon > 0$ and $g \in \mathcal{H}$ there is $m \leq \lceil 1/\epsilon^2 \rceil$ and $(f_i \in \mathcal{K}_i)_{i=1}^m$ and $(\gamma_i \in \mathbb{R})_{i=1}^m$, such that for every $w \in \mathcal{K}_{m+1}$*

$$\left| \left\langle w, g - (\sum_{i=1}^{m} \gamma_i f_i) \right\rangle \right| \leq \epsilon \ \|w\| \, \|g\| \, .$$

Finkelshtein et al. [11] introduced a version of Lemma D.1 (Lemma B.3 therein) with a "more constructive flavor." They provide a result in which the approximating vector $\sum_{i=1}^m \gamma_i f_i$ is given as the solution to a "manageable" optimization problem, whereas the original lemma in [43] only proves the existence of the approximating vector. Below, we give a similar result to [11, Lemma B.3.], where the constructive aspect is further improved. While Finkelshtein et al. [11] showed that the optimization problems leads to an approximate minimizer in high probability, they did not provide a way to evaluate if indeed this "good" event of high probability occurred. In contrast, we formulate this lemma in such a way that leads to a deterministic approach for checking whether the good event happened. In the discussion after Lemma D.2, we explain this in detail.

Lemma D.2. *Let $\{\mathcal{K}_j\}_{j\in\mathbb{N}}$ be a sequence of nonemply subsets of a real Hilbert space $\mathcal{H}$. Let $K \in \mathbb{N}$, $\delta \geq 0$, let $R \geq 1$ such that $K/R \in \mathbb{N}$, let $\delta > 0$, and let $g \in \mathcal{H}$. For every $k \in \mathbb{N}$, let*

$$\eta_k = (1+\delta) \inf_{\boldsymbol{\kappa}, \mathbf{h}} \|g - \sum_{i=1}^{k} \kappa_i h_i\|^2$$

where the infimum is over $\boldsymbol{\kappa} = \{\kappa_1, \ldots, \kappa_k\} \in \mathbb{R}^k$ and $\mathbf{h} = \{h_1, \ldots, h_k\} \in \mathcal{K}_1 \times \ldots \times \mathcal{K}_k$. Then,

1. *For every $m \in \mathbb{N}$, any vector of the form*

$$g^* = \sum_{j=1}^{m} \gamma_j f_j \quad \text{such that} \quad \boldsymbol{\gamma} = (\gamma_j)_{j=1}^m \in \mathbb{R}^m \quad \text{and} \quad \mathbf{f} = (f_j)_{j=1}^m \in \mathcal{K}_1 \times \ldots \times \mathcal{K}_m \tag{14}$$

   that gives a close-to-best Hilbert space approximation of g in the sense that

$$\|g - g^*\| \leq \eta_m, \tag{15}$$

   also satisfies

$$\forall w \in \mathcal{K}_{m+1}, \quad |\langle w, g - g^* \rangle| \leq \|w\| \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}}. \tag{16}$$

2. *If m is uniformly randomly sampled from $[K]$, then in probability $1 - \frac{1}{R}$ (with respect to the choice of m),*

$$\|w\| \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}} \leq \|w\| \, \|g\| \sqrt{\delta + \frac{R(1+\delta)}{K}}. \tag{17}$$

   Specifically, in probability $1 - \frac{1}{R}$, any vector of the form (14) which satisfy (15), also satisfies

$$\forall w \in \mathcal{K}_{m+1}, \quad |\langle w, g - g^* \rangle| \leq \|w\| \, \|g\| \sqrt{\delta + \frac{R(1+\delta)}{K}}. \tag{18}$$

The lemma is used as follows. We choose m at random. We know by Item 2 that in high probability the approximation is good (i.e. (18) is satisfied), but we are not certain. For certainty, we use the deterministic bound (16), which gives a certificate for a specific m. Namely, given a realization of m, we can estimate the right-hand-side of (16), which is also the left-hand-side of the probabilistic bound (17). For that, we find the $(m+1)$'th error η_{m+1}, solving another optimization problem, and verify that (17) is satisfied for m. If it is not, we resample m and repeat. The expected number of times we need to repeat this until we get a small error is $R/(R-1)$.

Hence, under an assumption that the we can find a close to optimum $\|g - g^*\|$ for a given m in T_K operations, we can find in probability 1 a vector g^* in the span of $\mathcal{K}_1, \ldots, \mathcal{K}_K$ that solves (18) with expected number of operations $T_K R/(R-1)$.

Proof of Lemma D.2. Let $K > 0$. Let $R \geq 1$ such that $K/R \in \mathbb{N}$. For every k, let

$$\eta_k = (1+\delta) \inf_{\boldsymbol{\kappa}, \mathbf{h}} \|g - \sum_{i=1}^{k} \kappa_i h_i\|^2$$

where the infimum is over $\boldsymbol{\kappa} = \{\kappa_1, \ldots, \kappa_k\} \in \mathbb{R}^k$ and $\mathbf{h} = \{h_1, \ldots, h_k\} \in \mathcal{K}_1 \times \ldots \times \mathcal{K}_k$.
Note that every

$$g^* = \sum_{j=1}^{m} \gamma_j f_j \tag{19}$$

that satisfies

$$\|g - g^*\|^2 \leq \eta_m$$

also satisfies: for any $w \in \mathcal{K}_{m+1}$ and every $t \in \mathbb{R}$,

$$\|g - (g^* + tw)\|^2 \geq \frac{\eta_{m+1}}{1+\delta} = \frac{\eta_m + \eta_{m+1} - \eta_m}{1+\delta} \geq \frac{\|g - g^*\|^2}{1+\delta} - \frac{\eta_m - \eta_{m+1}}{1+\delta}.$$

This can be written as

$$\forall t \in \mathbb{R}, \quad \|w\|^2 t^2 + 2\langle w, g - g^* \rangle t + \frac{\eta_m - \eta_{m+1}}{1+\delta} + (1 - \frac{1}{1+\delta}) \|g - g^*\|^2 \geq 0. \tag{20}$$

The discriminant of this quadratic polynomial is

$$4\langle w, g - g^* \rangle^2 - 4\|w\|^2 \left(\frac{\eta_m - \eta_{m+1}}{1+\delta} + (1 - \frac{1}{1+\delta}) \|g - g^*\|^2 \right)$$

and it must be non-positive to satisfy the inequality (20), namely

$$4\langle w, g - g^* \rangle^2 \leq 4\|w\|^2 \left(\frac{\eta_m - \eta_{m+1}}{1+\delta} + (1 - \frac{1}{1+\delta}) \|g - g^*\|^2 \right) \leq 4\|w\|^2 \left(\frac{\eta_m - \eta_{m+1}}{1+\delta} + (1 - \frac{1}{1+\delta}) \eta_m \right)$$

$$= 4\|w\|^2 \left(\eta_m - \frac{\eta_{m+1}}{1+\delta} \right),$$

which proves

$$|\langle w, g - g^* \rangle| \leq \|w\| \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}},$$

which proves Item 1.

For Item 2, note that $\|g\|^2 \geq \frac{\eta_1}{1+\delta} \geq \frac{\eta_2}{1+\delta} \geq \ldots \geq 0$. Therefore, there is a subset of at least $(1 - \frac{1}{R})K + 1$ indices m in $[K]$ such that $\eta_m \leq \eta_{m+1} + \frac{R(1+\delta)}{K} \|g\|^2$. Otherwise, there are $\frac{K}{R}$ indices m in $[K]$ such that $\eta_{m+1} < \eta_m - \frac{R(1+\delta)}{K} \|g\|^2$, which means that

$$\eta_K < \eta_1 - \frac{K}{R} \frac{R(1+\delta)}{K} \|g\|^2 \leq (1+\delta)\|g\|^2 - (1+\delta)\|g\|^2 = 0,$$

which is a contradiction to the fact that $\eta_K \geq 0$. Hence, there is a set $\mathcal{M} \subseteq [K]$ of $(1 - \frac{1}{R})K$ indices such that for every $m \in \mathcal{M}$,

$$\|w\| \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}} \leq \|w\| \sqrt{\eta_{m+1} + \frac{R(1+\delta)}{K} \|g\|^2 - \frac{\eta_{m+1}}{1+\delta}}$$

$$\leq \|w\| \sqrt{\frac{\delta}{1+\delta} \eta_{m+1} + \frac{R(1+\delta)}{K} \|g\|^2} \leq \|w\| \sqrt{\delta \|g\|^2 + \frac{R(1+\delta)}{K} \|g\|^2}$$

$$= \|w\| \|g\| \sqrt{\delta + \frac{R(1+\delta)}{K}}$$

□

D.3 The densifying semi-constructive graphon-signal weak regularity lemma

Define for kernel-signal (V, f) the *densifying cut distance*

$$\|(V,f)\|_{\square;q} = \frac{\alpha}{\iint q} \sup_{U,V} \left| \int_U \int_V V(x,y)q(x,y)dxdy \right| + \beta \frac{1}{D} \sum_{j=1}^{D} \sup_U \left| \int_U f_j(x)dx \right|.$$

Below we give a version of Theorem 4.1 for intersecting block graphons.

Theorem D.3. *Let (W, s) be a graphon-signal, $K \in \mathbb{N}$, $\delta > 0$, and let $\mathcal{Q}$ be a soft indicators model. Let q be a weight function and $\alpha, \beta > 0$. Let $R \geq 1$ such that $K/R \in \mathbb{N}$. Consider the graphon-signal Frobenius norm with weight $\|(Y,y)\|_{\mathrm{F};q} = \|(Y,y)\|_{\mathrm{F};q,\alpha,\beta}$, and cut norm with weight $\|(Y,y))\|_{\square;q} := \|(Y,y)\|_{\square;q,\alpha,\beta}$. For every $k \in \mathbb{N}$, let*

$$\eta_k = (1+\delta) \inf_{(C,p)\in[\mathcal{Q}]_k} \|(W,s) - (C,p)\|^2_{\mathrm{F};q}.$$

Then,

1. *For every $m \in \mathbb{N}$, any* IBG *$(C^*, p^*) \in [\mathcal{Q}]_m$ that gives a close-to-best weighted Frobenius approximation of (W, s) in the sense that*

$$\|(W,s) - (C^*,p^*)\|^2_{\mathrm{F};q} \leq \eta_m, \tag{21}$$

   also satisfies

$$\|(W,s) - (C^*,p^*)\|_{\square;q} \leq (\sqrt{\alpha} + \sqrt{\beta}) \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}}.$$

2. *If m is uniformly randomly sampled from $[K]$, then in probability $1 - \frac{1}{R}$ (with respect to the choice of m),*

$$\sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}} \leq \sqrt{\alpha \|W\|^2_{\mathrm{F};q} + \beta \|s\|^2_{\mathrm{F}}} \sqrt{\delta + \frac{R(1+\delta)}{K}}. \tag{22}$$

   Specifically, in probability $1 - \frac{1}{R}$, any $(C^*, p^*) \in [\mathcal{Q}]_m$ which satisfy (21), also satisfies

$$\|(W,s) - (C^*,p^*)\|_{\square;q} \leq (\sqrt{\alpha} + \sqrt{\beta}) \Big(\sqrt{\alpha \|W\|^2_{\mathrm{F},q} + \beta \|s\|^2_{\mathrm{F}}} \sqrt{\delta + \frac{R(1+\delta)}{K}}. \tag{23}$$

Theorem D.3 is similar to the semi-constructive weak regularity lemma of Finkelshtein et al. [11, Theorem B.1]. However, our result extends the result of Finkelshtein et al. [11] by providing a deterministic certificate for the approximation quality, as we explained in the discusson after Lemma D.2, extending the cut-norm to the more general weighted cut norm, and extending to general non-symmetric graphons.

Proof of Theorem D.3. Let us use Lemma D.2, with $\mathcal{H} = L^2([0,1]^2; q) \times (L^2[0,1])^D$ with the weighted inner product

$$\langle (V,y), (V',y') \rangle_q = \alpha \frac{1}{\|1\|_{1;q}} \iint_{[0,1]^2} V(x,y)V'(x,y)q(x,y)dxdy + \beta \sum_{j=1}^{D} \int_{[0,1]} y_j(x)y'_j(x)dx,$$

and corresponding norm denoted by $\|(Y,y)\|_{\mathrm{F};q} = \sqrt{\alpha \|Y\|^2_{\mathrm{F};q} + \beta \sum_{j=1}^{D} \|y_j\|^2_{\mathrm{F}}}$, and $\mathcal{K}_j = [\mathcal{Q}]$. Note that the Hilbert space norm is the Frobenius norm in this case. Let $m \in \mathbb{N}$. In the setting of the lemma, we take $g = (W, s)$, and $g^* \in [\mathcal{Q}]_m$. By the lemma, any approximate Frobenius minimizer (C^*, p^*), namely, that satisfies $\|(W,s) - (C^*,p^*)\|_{\mathrm{F};q} \leq \eta_m$, also satisfies

$$\langle (T,y), (W,s) - (C^*,p^*) \rangle_q \leq \|(T,y)\|_{\mathrm{F};q} \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}}$$

for every $(T, y) \in [\mathcal{Q}]$.

Hence, for every choice of measurable subsets $\mathcal{S}, \mathcal{T} \subset [0,1]$, we have

$$\begin{aligned}
&\frac{1}{\|1\|_{1;q}} \left| \int_{\mathcal{S}} \int_{\mathcal{T}} (W(x,y) - C^*(x,y)) q(x,y) dx dy \right| \\
&= \left| \frac{1}{\alpha} \left\langle (\mathbb{1}_{\mathcal{S}} \otimes \mathbb{1}_{\mathcal{T}}, 0), (W,s) - (C^*, p^*) \right\rangle_q \right| \\
&\leq \frac{1}{\alpha} \|(\mathbb{1}_{\mathcal{S}} \otimes \mathbb{1}_{\mathcal{T}}, 0)\|_{\mathrm{F};q} \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}} \\
&\leq \frac{1}{\alpha} \sqrt{\alpha} \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}}
\end{aligned}$$

Hence, taking the supremum over $\mathcal{S}, \mathcal{T} \subset [0,1]$, we also have

$$\alpha \|W - C^*\|_{\square;q} \leq \sqrt{\alpha} \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}}.$$

Now, for n randomly uniformly sampled from $[K]$, consider the event $\mathcal{M}$ (regarding the uniform choice of n) of probability $(1 - 1/R)$ in which

$$\sqrt{\eta_n - \frac{\eta_{n+1}}{1+\delta}} \leq \sqrt{\alpha \|W\|_{\mathrm{F};q}^2 + \beta \|s\|_{\mathrm{F}}^2} \sqrt{\delta + \frac{R(1+\delta)}{K}}.$$

Hence, in the event $\mathcal{M}$, we also have

$$\alpha \|W - C^*\|_{\square;q} \leq \sqrt{\alpha^2 \|W\|_{\mathrm{F};q}^2 + \alpha\beta \|s\|_{\mathrm{F}}} \sqrt{\delta + \frac{R(1+\delta)}{K}}.$$

Similarly, for every measurable $\mathcal{T} \subset [0,1]$ and every standard basis element $\boldsymbol{b} = (\delta_{j,i})_{i=1}^D$ for any $j \in [D]$,

$$\begin{aligned}
&\left| \int_{\mathcal{T}} (s_j(x) - p_j^*(x)) dx \right| \\
&= \left| \frac{1}{\beta} \left\langle (0, \boldsymbol{b}\mathbb{1}_{\mathcal{T}}), (W,s) - (C^*, p^*) \right\rangle_q \right| \\
&\leq \frac{1}{\beta} \|(0, \boldsymbol{b}\mathbb{1}_{\mathcal{T}})\|_{\mathrm{F};q} \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}} \\
&\leq \frac{\sqrt{\beta}}{\beta} \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}},
\end{aligned}$$

so, taking the supremum over $\mathcal{T} \subset [0,1]$ independently for every $j \in [D]$, and averaging over $j \in [D]$, we get

$$\beta \|s - p^*\|_{\square} \leq \sqrt{\beta} \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}}.$$

Now, for the same event $\mathcal{M}$ as above regarding the choice of $n \in [K]$,

$$\beta \|s - p^*\|_{\square} \leq \sqrt{\alpha\beta \|W\|_{\mathrm{F};q}^2 + \beta^2 \|s\|_{\mathrm{F}}^2} \sqrt{\delta + \frac{R(1+\delta)}{K}}.$$

Overall, we get for every m and corresponding approximately optimum (C^*, p^*),

$$\|(W,s) - (C^*, p^*)\|_{\square;q} \leq (\sqrt{\alpha} + \sqrt{\beta}) \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}}.$$

Moreover, for uniformly sampled $n \in [K]$, in probability more than $1 - 1/R$,

$$\|(W,s) - (C^*, p^*)\|_{\square;q} \leq (\sqrt{\alpha} + \sqrt{\beta}) \Big(\sqrt{\alpha \|W\|_{\mathrm{F},q}^2 + \beta \|s\|_{\mathrm{F}}^2} \sqrt{\delta + \frac{R(1+\delta)}{K}}.$$

□

D.4 Proof of the semi-constructive densifying weak regularity lemma

Next, we show that Theorem D.3 reduces to Theorem 4.1 in the case of graphon-signals induced by graph-signals.

Theorem 4.1. *Let $(\boldsymbol{A}, \boldsymbol{X})$ be a graph-signal, $K \in \mathbb{N}$, $\delta > 0$, and let $\mathcal{Q}$ be a soft indicators model. Let $\alpha, \beta > 0$ such that $\alpha+\beta = 1$. Let $\Gamma > 0$ and let $\boldsymbol{Q}_{\boldsymbol{A}}$ be the weight matrix defined in Definition 3.2. Let $R \geq 1$ such that $K/R \in \mathbb{N}$. For every $k \in \mathbb{N}$, let*

$$\eta_k = (1+\delta) \min_{(\boldsymbol{C},\boldsymbol{P})\in[\mathcal{Q}]_k} \|(\boldsymbol{A}, \boldsymbol{X}) - (\boldsymbol{C}, \boldsymbol{P})\|^2_{\mathrm{F};\boldsymbol{Q}_{\boldsymbol{A}},\alpha(1+\Gamma),\beta}$$

Then,

1. *For every $m \in \mathbb{N}$, any* IBG $(\boldsymbol{C}^*, \boldsymbol{P}^*) \in [\mathcal{Q}]_m$ *that gives a close-to-best weighted Frobenius approximation of $(\boldsymbol{A}, \boldsymbol{X})$ in the sense that*

$$\|(\boldsymbol{A}, \boldsymbol{X}) - (\boldsymbol{C}^*, \boldsymbol{P}^*)\|^2_{\mathrm{F};\boldsymbol{Q}_{\boldsymbol{A}},\alpha(1+\Gamma),\beta} \leq \eta_m, \tag{24}$$

   also satisfies

$$\sigma_{\square;\alpha,\beta,\Gamma}\big((\boldsymbol{A}, \boldsymbol{X})||(\boldsymbol{C}^*, \boldsymbol{P}^*)\big) \leq (\sqrt{\alpha(1+\Gamma)} + \sqrt{\beta})\sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}}. \tag{25}$$

2. *If m is uniformly randomly sampled from $[K]$, then in probability $1 - \frac{1}{R}$ (with respect to the choice of m),*

$$\sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}} \leq \sqrt{\delta + \frac{R(1+\delta)}{K}} \tag{26}$$

   Specifically, in probability $1 - \frac{1}{R}$, any $(\boldsymbol{C}^*, \boldsymbol{P}^*) \in [\mathcal{Q}]_m$ which satisfy (24), also satisfies

$$\sigma_{\square;\alpha,\beta,\Gamma}\big((\boldsymbol{A}, \boldsymbol{X}) - (\boldsymbol{C}^*, \boldsymbol{P}^*)\big) \leq \big(\sqrt{2+\Gamma}\big)\sqrt{\delta + \frac{R(1+\delta)}{K}}. \tag{27}$$

In practice, Theorem 4.1 is used to motivate the following computational approach for approximating graph-signals by IBGs. We suppose that there is an oracle optimization method that can solve (24) in T_K operations whenever $m \leq K$. In practice, we use gradient descent on the left-hand side of (24), which takes $O(E)$ operations as shown in Proposition 4.2. The oracle is used as follows. We choose $m \in [K]$ at random. We know by Item 2 of Theorem 4.1 that in high probability the good approximation bound (27) is satisfied, but we are not certain. For certainty, we use Item 1 of Theorem 4.1. Given our specific realization of m, we can estimate the right-hand-side of (25) by our oracle optimization method in $2T_K$ operations, and verify that the right-hand-side of (25) is less than the right-hand-side of (27). If it is not (in probability $1/R$), we resample m and repeat. The expected number of times we need to repeat this process until we get a small error is $R/(R-1)$, so the expected time it takes the algorithm to find an IBG with error bound (27) is $2T_K R/(R-1)$.

Proof of Theorem 4.1. Let $\boldsymbol{Q}_{\boldsymbol{A}}$ be the weight matrix defined in (12). Consider the following identities between the weighted Frobenius norm of induced graphon-signals and the Frobenius norm of the graph-signal, and, a similar identity for the densifying cut similarity.

$$\begin{aligned}
&\|(W_{\boldsymbol{A}}, s_{\boldsymbol{X}}) - (W_{\boldsymbol{C}}, W_{\boldsymbol{P}})\|^2_{\mathrm{F};W_{\boldsymbol{Q}_{\boldsymbol{A}}},\alpha(1+\Gamma),\beta} = \|(\boldsymbol{A}, \boldsymbol{X}) - (\boldsymbol{C}, \boldsymbol{P})\|^2_{\mathrm{F};\boldsymbol{Q}_{\boldsymbol{A}},\alpha(1+\Gamma),\beta}\,, \\
&\sigma_{\square,\alpha,\beta}\big((W_{\boldsymbol{A}}, s_{\boldsymbol{X}})||(W_{\boldsymbol{C}}, s_{\boldsymbol{P}})\big) = \sigma_{\square,\alpha,\beta}\big((\boldsymbol{A}, \boldsymbol{X})||(\boldsymbol{C}, \boldsymbol{P})\big).
\end{aligned} \tag{28}$$

We apply Theorem D.3 on the weighted Frobenius and cut norms with weight $Q = W_{\boldsymbol{Q}_{\boldsymbol{A}}}$. We immediately obtain (25) from (28). For (26), by (22) of Theorem D.3, and by (13) and by the fact that signals have values in $[-1, 1]$,

$$\begin{aligned}
\sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}} &\leq \sqrt{\alpha(1+\Gamma)\|W_{\boldsymbol{A}}\|^2_{\mathrm{F};W_{\boldsymbol{Q}_{\boldsymbol{A}}}} + \beta\,\|s_{\boldsymbol{A}}\|^2_{\mathrm{F}}}\sqrt{\delta + \frac{R(1+\delta)}{K}} \\
&\leq \sqrt{\delta + \frac{R(1+\delta)}{K}}.
\end{aligned}$$

Lastly, (27) follows the fact that for $\alpha, \beta > 0$ such that $\alpha + \beta = 1$, we must have $(\sqrt{\alpha(1+\Gamma)} + \sqrt{\beta}) \leq \sqrt{2+\Gamma}$. □

E Fitting IBGs to graphs efficiently

Below we present the proof of Proposition 4.2. The proof follows the lines of the proof of Proposition 4.1 in [11]. We restate the proposition below for the benefit of the reader.

Proposition 4.2. *Let $\boldsymbol{A} = (a_{i,j})_{i,j=1}^{N}$ be an adjacency matrix of an unweighted graph with E edges. The graph part of the sparse Frobenius loss can be written as*

$$\begin{aligned}\left\|\boldsymbol{A} - \boldsymbol{U}\operatorname{diag}(\boldsymbol{r})\boldsymbol{V}^\top\right\|^2_{\mathrm{F};\boldsymbol{Q_A}} = \|A\|^2_{\mathrm{F};\boldsymbol{Q_A}} + \frac{e}{(1+\Gamma)E}\operatorname{Tr}\left((\boldsymbol{V}^\top\boldsymbol{V})\operatorname{diag}(\boldsymbol{r})(\boldsymbol{U}^\top\boldsymbol{U})\operatorname{diag}(\boldsymbol{r})\right)\\ - \frac{2}{(1+\Gamma)E}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}(i)}\boldsymbol{U}_{i,:}\operatorname{diag}(\boldsymbol{r})\left(\boldsymbol{V}^\top\right)_{:,j}a_{i,j}\\ + \frac{1-e}{(1+\Gamma)E}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}(i)}(\boldsymbol{U}_{i,:}\operatorname{diag}(\boldsymbol{r})\left(\boldsymbol{V}^\top\right)_{:,j})^2\end{aligned}$$

where $\boldsymbol{Q_A}$ is defined in Equation (2)*. Computing the right-hand-side and its gradients with respect to $\boldsymbol{U}$, $\boldsymbol{V}$ and $\boldsymbol{r}$ has a time complexity of $\mathcal{O}(K^2N + KE)$, and a space complexity of $\mathcal{O}(KN + E)$.*

The proof is similar to that of Proposition 4.1 in [11], while applying the necessary changes under the new weighted Frobenius norm and the structure of IBGs.

Proof. The loss can be expressed as

$$\begin{aligned}\left\|\boldsymbol{A} - \boldsymbol{U}\operatorname{diag}(\boldsymbol{r})\boldsymbol{V}^\top\right\|^2_{\mathrm{F};\boldsymbol{Q_A}} &= \frac{1}{(1+\Gamma)\,E}\sum_{i=1}^{N}\left(\sum_{j\in\mathcal{N}(i)}\left(a_{i,j} - \boldsymbol{U}_{i,:}\operatorname{diag}(\boldsymbol{r})\left(\boldsymbol{V}^\top\right)_{:,j}\right)^2 + \right.\\ &\qquad\left. + \sum_{j\notin\mathcal{N}(i)} e\left(-\boldsymbol{U}_{i,:}\operatorname{diag}(\boldsymbol{r})\left(\boldsymbol{V}^\top\right)_{:,j}\right)^2\right)\\ &= \frac{1}{(1+\Gamma)\,E}\sum_{i=1}^{N}\left(\sum_{j\in\mathcal{N}(i)}\left(a_{i,j} - \boldsymbol{U}_{i,:}\operatorname{diag}(\boldsymbol{r})\left(\boldsymbol{V}^\top\right)_{:,j}\right)^2 + \sum_{j=1}^{N} e\left(\boldsymbol{U}_{i,:}\operatorname{diag}(\boldsymbol{r})\left(\boldsymbol{V}^\top\right)_{:,j}\right)^2\right.\\ &\qquad\left. - \sum_{j\in\mathcal{N}(i)} e\left(\boldsymbol{U}_{i,:}\operatorname{diag}(\boldsymbol{r})\left(\boldsymbol{V}^\top\right)_{:,j}\right)^2\right)\end{aligned}$$

We expand the quadratic term $\left(a_{i,j} - \boldsymbol{U}_{i,:}\operatorname{diag}(\boldsymbol{r})\left(\boldsymbol{V}^\top\right)_{:,j}\right)^2$, and get

$$\begin{aligned}\left\|\boldsymbol{A} - \boldsymbol{U}\operatorname{diag}(\boldsymbol{r})\boldsymbol{V}^\top\right\|^2_{\mathrm{F};\boldsymbol{Q_A}} &= \frac{e}{(1+\Gamma)\,E}\sum_{i,j=1}^{N}\left(\boldsymbol{U}_{i,:}\operatorname{diag}(\boldsymbol{r})\left(\boldsymbol{V}^\top\right)_{:,j}\right)^2 +\\ &\quad + \frac{1}{(1+\Gamma)\,E}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}(i)}\left(a_{i,j}^2 - 2\boldsymbol{U}_{i,:}\operatorname{diag}(\boldsymbol{r})\left(\boldsymbol{V}^\top\right)_{:,j}a_{i,j}\right)\\ &\quad + \frac{1-e}{(1+\Gamma)\,E}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}(i)}\left(\boldsymbol{U}_{i,:}\operatorname{diag}(\boldsymbol{r})\left(\boldsymbol{V}^\top\right)_{:,j}\right)^2\\ &= \frac{eN^2}{(1+\Gamma)\,E}\left\|\boldsymbol{U}\operatorname{diag}(\boldsymbol{r})\boldsymbol{V}^\top\right\|^2_{\mathrm{F}} +\\ &\quad + \frac{1}{(1+\Gamma)\,E}\sum_{i,j=1}^{N}a_{i,j}^2 - \frac{2}{(1+\Gamma)\,E}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}(i)}\boldsymbol{U}_{i,:}\operatorname{diag}(\boldsymbol{r})\left(\boldsymbol{V}^\top\right)_{:,j}a_{i,j}\\ &\quad + \frac{1-e}{(1+\Gamma)\,E}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}(i)}\left(\boldsymbol{U}_{i,:}\operatorname{diag}(\boldsymbol{r})\left(\boldsymbol{V}^\top\right)_{:,j}\right)^2\end{aligned}$$

$$
\begin{aligned}
&= \frac{e}{(1+\Gamma)\,E}\,\mathrm{Tr}\left(\boldsymbol{V}^\top\boldsymbol{V}\,\mathrm{diag}(\boldsymbol{r})\boldsymbol{U}^\top\boldsymbol{U}\,\mathrm{diag}(\boldsymbol{r})\right)+\\
&+\frac{N^2}{(1+\Gamma)\,E}\,\|\boldsymbol{A}\|_{\mathrm{F}}^2\\
&-\frac{2}{(1+\Gamma)\,E}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}(i)}\boldsymbol{U}_{i,:}\,\mathrm{diag}(\boldsymbol{r})\left(\boldsymbol{V}^\top\right)_{:,j}a_{i,j}+\\
&\frac{1-e}{(1+\Gamma)\,E}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}(i)}\left(\boldsymbol{U}_{i,:}\,\mathrm{diag}(\boldsymbol{r})\left(\boldsymbol{V}^\top\right)_{:,j}\right)^2
\end{aligned}
$$

Here, the last equality uses the trace cyclicity, i.e., $\forall \boldsymbol{I},\boldsymbol{J}\in\mathbb{R}^{N\times K}:\ \mathrm{Tr}(\boldsymbol{I}\boldsymbol{J}^\top)=\mathrm{Tr}(\boldsymbol{J}^\top\boldsymbol{I})$, with $\boldsymbol{I}=\boldsymbol{V}\,\mathrm{diag}(\boldsymbol{r})\boldsymbol{U}^\top\boldsymbol{U}\,\mathrm{diag}(\boldsymbol{r})$ and $\boldsymbol{J}^\top=\boldsymbol{V}^\top$.

To calculate the first term efficiently, we can either perform matrix multiplication from right to left or compute $\boldsymbol{U}^\top\boldsymbol{U}$ and $\boldsymbol{V}^\top\boldsymbol{V}$, followed by the rest of the product. This calculation has a time complexity of $\mathcal{O}(K^2N)$ and a memory complexity $\mathcal{O}(KN)$. The second term in the equality is constant and, therefore, can be left out during optimization. The third and fourth terms in the expression are calculated using message-passing, and thus have a time complexity of $\mathcal{O}(KE)$. Overall, we end up with a complexity of $\mathcal{O}(K^2N+KE)$ and a space complexity of $\mathcal{O}(KN+E)$ for the full computation of the loss and its gradients with respect to $\boldsymbol{U}$, $\boldsymbol{V}$ and $\boldsymbol{r}$. □

F Extending the densifying weak regularity lemma for graphon-edge-signals

In this section we prove a version of Theorem D.3 for the case where the graph has an edge signal. The proof is very similar to the previous case, and the new theorem can be used for the analysis of IBG-NN when used for knowledge graphs (see Appendix I).

F.1 Weighted Frobenius and cut norms for graphon-edge-signals

Graphon-edge-signal A graphon-edge-signal is a pair (V,Y) where V is a graphon and $Y:[0,1]^2\to\mathbb{R}^D$ is a measurable function.

Weighted edge-signal Frobenius norm Consider the real Hilbert space $L^2([0,1]^2;q)\times(L^2[0,1]^2)^D$ defined with the weighted inner product

$$
\begin{aligned}
&\langle(V,Y),(V',Y')\rangle_q=\langle(V,Y),(V',Y')\rangle_{q,\alpha,\beta}=\\
&=\alpha\frac{1}{\|1\|_{1;q}}\iint_{[0,1]^2}V(x,y)V'(x,y)q(x,y)dxdy+\frac{\beta}{D}\sum_{j=1}^{D}\iint_{[0,1]^2}Y_j(x,y)Y_j'(x,y)dxdy.
\end{aligned}
$$

We call the corresponding weighted norm the *weighted edge-signal Frobenius norm*, denoted by

$$
\|(V,Y)\|_{\mathrm{F};q}=\|(V,Y)\|_{\mathrm{F};q,\alpha,\beta}=\sqrt{\alpha\,\|V\|_{\mathrm{F};q}^2+\frac{\beta}{D}\sum_{j=1}^{D}\|Y_j\|_{\mathrm{F}}^2},
$$

We similarly extend the definition of a graphon weighted cut norm and cut metric.

Graphon weighted cut norm and cut metric. A kernel-edge-signal (V,Y) is a pair where $V:[0,1]^2\to[-1,1]$ and $Y:[0,1]^2\to\mathbb{R}^D$ are measurable. Define for a kernel-edge-signal (V,Y) the *weighted edge signal cut norm*

$$
\begin{aligned}
\|(V,Y)\|_{\square;q,\alpha,\beta}=\|(V,Y)\|_{\square;q}=\frac{\alpha}{\|1\|_{1;q}}\sup_{\mathcal{U},\mathcal{V}}\left|\int_{\mathcal{U}}\int_{\mathcal{V}}V(x,y)q(x,y)dxdy\right|+\\
+\beta\frac{1}{D}\sum_{j=1}^{D}\sup_{\mathcal{U},\mathcal{V}}\left|\int_{\mathcal{U}}\int_{\mathcal{V}}Y_j(x,y)dxdy\right|,
\end{aligned}
$$

where the supremum is over the set of measurable subsets $\mathcal{U}, \mathcal{V} \subset [0,1]$.

The weighted edge signal cut metric between two graphon-edge-signals (W, f) and (W', f') is defined to be $\|(W, f) - (W', f')\|_{\square;q}$.

For simplicity's sake, and for this section only, we refer to the weighted edge-signal Frobenius and cut norms simply as the weighted Frobenius and cut norms.

F.2 IBGs with edge signals

Here, we define IBGs for graphon-edge-signals. We use the same terminology of *soft rank-K* IBG *model* introduced in Definition D.2, slightly changing the signal part of the graphon.

Definition F.1. *Let $D \in \mathbb{N}$. Given a soft affiliation model $\mathcal{Q}$, the subset $[\mathcal{Q}]$ of $L^2[0,1]^2 \times (L^2[0,1]^2)^D$ of all elements of the form $(au(x)v(y), bu(z)v(w))$, with $u, v \in \mathcal{Q}$, $a \in \mathbb{R}$ and $b \in \mathbb{R}^D$, is called the* soft rank-1 intersecting block graphon (IBG) model *corresponding to $\mathcal{Q}$. Given $K \in \mathbb{N}$, the subset $[\mathcal{Q}]_K$ of $L^2[0,1]^2 \times (L^2[0,1]^2)^D$ of all linear combinations of K elements of $[\mathcal{Q}]$ is called the* soft rank-K IBG model *corresponding to $\mathcal{Q}$. Namely, $(C, p) \in [\mathcal{Q}]_K$ if and only if it has the form*

$$C(x,y) = \sum_{k=1}^{K} a_k u_k(x) v_k(y) \quad \textit{and} \quad p(x,y) = \sum_{k=1}^{K} b_k u_k(x) v_k(y)$$

where $(u_k)_{k=1}^K \in \mathcal{Q}^K$ are called the target community affiliation functions*, $(v_k)_{k=1}^K \in \mathcal{Q}^K$ are called the* source community affiliation functions*, $(a_k)_{k=1}^K \in \mathbb{R}^K$ are called the* community affiliation magnitudes*, $(b_k)_{k=1}^K \in \mathbb{R}^{K \times D}$ are called the* edge features*. Any element of $[\mathcal{Q}]_K$ is called an intersecting block graphon-signal (*IBG*).*

We emphasize that for the rest of this section, when referencing weighted Frobenius and cut norms, as well as the *soft rank-K* IBG *model*, we refer to the new definitions as formulated in Appendix F.

Corollary F.1. *Let (W, s) be a graphon-edge-signal, $K \in \mathbb{N}$, $\delta > 0$, and let $\mathcal{Q}$ be a soft indicators model. Let q be a weight function and $\alpha, \beta > 0$. Let $R \geq 1$ such that $K/R \in \mathbb{N}$. Consider the graphon-signal Frobenius norm with weight $\|(Y,y)\|_{\mathrm{F};q} = \|(Y,y)\|_{\mathrm{F};q,\alpha,\beta}$, and cut norm with weight $\|(Y,y)\|_{\square;q} := \|(Y,y)\|_{\square;q,\alpha,\beta}$. For every $k \in \mathbb{N}$, let*

$$\eta_k = (1+\delta) \inf_{(C,p) \in [\mathcal{Q}]_k} \|(W,s) - (C,p)\|_{\mathrm{F};q}^2.$$

Then,

1. *For every $m \in \mathbb{N}$, any* IBG *$(C^*, p^*) \in [\mathcal{Q}]_m$ that gives a close-to-best weighted Frobenius approximation of (W, s) in the sense that*

$$\|(W,s) - (C^*,p^*)\|_{\mathrm{F};q}^2 \leq \eta_m, \tag{29}$$

also satisfies

$$\|(W,s) - (C^*,p^*)\|_{\square;q} \leq (\sqrt{\alpha} + \sqrt{\beta}) \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}}.$$

2. *If m is uniformly randomly sampled from $[K]$, then in probability $1 - \frac{1}{R}$ (with respect to the choice of m),*

$$\sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}} \leq \sqrt{\alpha \|W\|_{\mathrm{F};q}^2 + \beta \|s\|_{\mathrm{F}}^2} \sqrt{\delta + \frac{R(1+\delta)}{K}}. \tag{30}$$

Specifically, in probability $1 - \frac{1}{R}$, any $(C^*, p^*) \in [\mathcal{Q}]_m$ which satisfy (29), also satisfies

$$\|(W,s) - (C^*,p^*)\|_{\square;q} \leq (\sqrt{\alpha} + \sqrt{\beta}) \Big(\sqrt{\alpha \|W\|_{\mathrm{F},q}^2 + \beta \|s\|_{\mathrm{F}}^2} \sqrt{\delta + \frac{R(1+\delta)}{K}}.$$

The proof is very similar to the original proof, with a slight adjustment for the analysis of the signal part of the graphon-signal. For completeness of the analysis, we provide the full proof.

Proof. Let us use Lemma D.2, with $\mathcal{H} = L^2([0,1]^2; q) \times (L^2[0,1]^2)^D$ with the weighted inner product

$$\langle (V,Y),(V',Y')\rangle_q = \alpha \frac{1}{\|1\|_{1;q}} \iint_{[0,1]^2} V(x,y)V'(x,y)q(x,y)dxdy + \\ + \beta \sum_{j=1}^{D} \frac{1}{\|1\|_{1;q}} \iint_{[0,1]^2} Y(x,y)Y'(x,y)q(x,y)dxdy,$$

and corresponding norm denoted by $\|(V,Y)\|_{\mathrm{F};q} \sqrt{\alpha \|V\|^2_{\mathrm{F};q} + \beta \sum_{j=1}^{D} \|Y_j\|^2_{\mathrm{F};q}}$, and $\mathcal{K}_j = [\mathcal{Q}]$. Note that the Hilbert space norm is a weighted Frobenius norm. Let $m \in \mathbb{N}$. In the setting of the lemma, we take $g = (W,s)$, and $g^* \in [\mathcal{Q}]_m$. By the lemma, any approximate Frobenius minimizer (C^*, p^*), namely, that satisfies $\|(W,s) - (C^*,p^*)\|_{\mathrm{F};q} \le \eta_m$, also satisfies

$$\langle (T,y),(W,s)-(C^*,p^*)\rangle_q \le \|(T,y)\|_{\mathrm{F};q} \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}}$$

for every $(T,y) \in [\mathcal{Q}]$. Hence, for every choice of measurable subsets $\mathcal{S}, \mathcal{T} \subset [0,1]$, we have

$$\begin{aligned} &\frac{1}{\|1\|_{1;q}} \left| \int_{\mathcal{S}} \int_{\mathcal{T}} (W(x,y) - C^*(x,y))q(x,y)dxdy \right| \\ &= \left| \frac{1}{\alpha} \langle (\mathbb{1}_{\mathcal{S}} \otimes \mathbb{1}_{\mathcal{T}}, 0), (W,s) - (C^*,p^*)\rangle_q \right| \\ &\le \frac{1}{\alpha} \|(\mathbb{1}_{\mathcal{S}} \otimes \mathbb{1}_{\mathcal{T}}, 0)\|_{\mathrm{F};q} \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}} \\ &\le \frac{1}{\alpha} \sqrt{\alpha} \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}} \end{aligned}$$

Hence, taking the supremum over $\mathcal{S}, \mathcal{T} \subset [0,1]$, we also have

$$\alpha \|W - C^*\|_{\square;q} \le \sqrt{\alpha} \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}}.$$

Now, for n randomly uniformly from $[K]$, consider the event $\mathcal{M}$ (regarding the uniform choice of n) of probability $(1 - 1/R)$ in which

$$\sqrt{\eta_n - \frac{\eta_{n+1}}{1+\delta}} \le \sqrt{\alpha \|W\|^2_{\mathrm{F};q} + \beta \|s\|^2_{\mathrm{F}}} \sqrt{\delta + \frac{R(1+\delta)}{K}}.$$

Hence, in the event $\mathcal{M}$, we also have

$$\alpha \|W - C^*\|_{\square;q} \le \sqrt{\alpha^2 \|W\|^2_{\mathrm{F};q} + \alpha\beta \|s\|_{\mathrm{F}}} \sqrt{\delta + \frac{R(1+\delta)}{K}}.$$

Similarly, for every measurable $\mathcal{S}, \mathcal{T} \subset [0,1]$ and every standard basis element $\boldsymbol{b} = (\delta_{j,i})_{i=1}^{D}$ for any $j \in [D]$,

$$\begin{aligned} &\frac{1}{\|1\|_{1;q}} \left| \int_{\mathcal{S}} \int_{\mathcal{T}} (s(x,y) - p^*(x,y))q(x,y)dxdy \right| \\ &= \left| \frac{1}{\beta} \langle (0, \boldsymbol{b}(\mathbb{1}_{\mathcal{S}} \otimes \mathbb{1}_{\mathcal{T}}), (W,s) - (C^*,p^*)\rangle_q \right| \\ &\le \frac{1}{\beta} \|(0, \boldsymbol{b}(\mathbb{1}_{\mathcal{S}} \otimes \mathbb{1}_{\mathcal{T}})\|_{\mathrm{F};q} \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}} \\ &\le \frac{1}{\beta} \sqrt{\beta} \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}} \end{aligned}$$

so, taking the supremum over $\mathcal{S}, \mathcal{T} \subset [0,1]$, independently for every $j \in [D]$, and averaging over $j \in [D]$, we get

$$\beta \|s - p^*\|_{\square;q} \le \sqrt{\beta} \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}}.$$

Now, for the same event $\mathcal{M}$ as above regarding the choice of $n \in [K]$,

$$\beta \left\| s - p^* \right\|_\square \leq \sqrt{\alpha\beta \|W\|_{\mathrm{F};q}^2 + \beta^2 \left\| s \right\|_{\mathrm{F}}^2} \sqrt{\delta + \frac{R(1+\delta)}{K}}.$$

Overall, we get for every m and corresponding approximately optimum (C^*, p^*),

$$\|(W, s) - (C^*, p^*)\|_{\square;q} \leq (\sqrt{\alpha} + \sqrt{\beta}) \sqrt{\eta_m - \frac{\eta_{m+1}}{1+\delta}}.$$

Moreover, for uniformly sampled $n \in [K]$, in probability more than $1 - 1/R$,

$$\|(W, s) - (C^*, p^*)\|_{\square;q} \leq (\sqrt{\alpha} + \sqrt{\beta}) \Big(\sqrt{\alpha \|W\|_{\mathrm{F},q}^2 + \beta \left\| s \right\|_{\mathrm{F}}^2} \sqrt{\delta + \frac{R(1+\delta)}{K}}.$$

□

G Initializing the optimization with singular vectors

Here, we propose a good initialization for the GD minimization of 8. We explain how to use the SVD of the graph to initialize the parameters of a rank K-IBG, before the gradient descent minimization of Equation (8). This is inspired by the eigendecomposition initialization described of ICGs. The full method is presented in Appendix G, and summarized here.

We begin by calculating the $K/4$ SVD decomposition of the graph adjacency matrix. Denote by $\boldsymbol{\sigma}_{K/4} = (\sigma_k)_{k=1}^{K/4}$ the sequence of the $K/4$ largest singular values of $\boldsymbol{A}$, and by $\boldsymbol{\Phi}_{K/4} = (\boldsymbol{\phi}_k)_{k=1}^{K/4}$, $\boldsymbol{\Psi}_{K/4} = (\boldsymbol{\psi}_k)_{k=1}^{K/4}$ their corresponding left and right singular vectors.

For each singular value σ and corresponding singular vectors $\boldsymbol{\phi}, \boldsymbol{\psi}$, we designate $\{\frac{\boldsymbol{\phi}_+}{\|\boldsymbol{\phi}_+\|}, \frac{\boldsymbol{\phi}_+}{\|\boldsymbol{\phi}_+\|}, \frac{\boldsymbol{\phi}_-}{\|\boldsymbol{\phi}_-\|}, \frac{\boldsymbol{\phi}_-}{\|\boldsymbol{\phi}_-\|}\}$ as target communities, and $\{\frac{\boldsymbol{\psi}_+}{\|\boldsymbol{\psi}_+\|}, \frac{\boldsymbol{\psi}_-}{\|\boldsymbol{\psi}_-\|}, \frac{\boldsymbol{\psi}_+}{\|\boldsymbol{\psi}_+\|}, \frac{\boldsymbol{\psi}_-}{\|\boldsymbol{\psi}_-\|}\}$ as the corresponding source communities, where $\boldsymbol{\xi}_\pm \in [0, \infty)^N$ denotes the positive or negative parts of the vector $\boldsymbol{\xi}$, i.e., $\boldsymbol{\xi} = \boldsymbol{\xi}_+ - \boldsymbol{\xi}_-$. The corresponding affiliation magnitudes are then taken to be

$$r_1 = \sigma \left\| \boldsymbol{\phi}_+ \right\|_\infty \left\| \boldsymbol{\psi}_+ \right\|_\infty, \; r_2 = -\sigma \left\| \boldsymbol{\phi}_+ \right\|_\infty \left\| \boldsymbol{\psi}_- \right\|_\infty,$$
$$r_3 = -\sigma \left\| \boldsymbol{\phi}_- \right\|_\infty \left\| \boldsymbol{\psi}_+ \right\|_\infty, r_4 = \sigma \left\| \boldsymbol{\phi}_- \right\|_\infty \left\| \boldsymbol{\psi}_- \right\|_\infty.$$

If K is not divisible by 4, we discard the excess components with the smallest community affiliation magnitudes in absolute value.

To efficiently calculate the leading left and right singular vectors, we may use power method variants such as the Lanczos algorithm [44] or simultaneous iteration [45] in $\mathcal{O}(E)$ operations per iteration. For very large graphs, we propose in Appendix G.2 a more efficient randomized SVD algorithm that does not require reading the whole graph into memory at once.

G.1 Theoretical analysis of SVD initialization

Next, we analyze this initialization and show that it attains a relatively high initial accuracy. The following is a corollary of the densifying weak regularity lemma with constant $q(x, y) = \frac{N^2}{E}$, the signal weight in the cut norm set to $\beta = 0$, using all measurable functions from $[0, 1]^2$ to $[0, 1]$ as the soft affiliation model, and taking relative Frobenius error with $\delta = 0$ on theorem D.3. In this case, according to the best rank-K approximation theorem (Eckart–Young–Mirsky Theorem [45, Theorem 5.9]), the minimizer of the Frobenius error is the rank-K singular value decomposition (SVD). This leads to the following corollary.

Corollary G.1. *Let $\boldsymbol{A}$ be a graph, $K \in \mathbb{N}$, let m be sampled uniformly from $[K]$, and let $R \geq 1$ such that $K/R \in \mathbb{N}$. Let $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m$ and $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m$ be the leading left and right singular vectors of $\boldsymbol{A}$ respectively, with singular values $\sigma_1, \ldots, \sigma_m$ of highest magnitudes $|\sigma_1| \geq |\sigma_2| \geq \ldots \geq |\sigma_m|$, and let $\boldsymbol{C}^* = \sum_{k=1}^m \sigma_k \boldsymbol{u}_k \boldsymbol{v}_k^\top$. Then, in probability $1 - \frac{1}{R}$ (with respect to the choice of m),*

$$\left\| \boldsymbol{A} - \boldsymbol{C}^* \right\|_\square < \left\| \boldsymbol{A} \right\|_{\mathrm{F}} \sqrt{\frac{R}{K}}.$$

Proof. Consider Theorem D.3, with $\delta = 0, \beta = 0, \Gamma = 0$, and taking the constant weight

$$q(x, y) = \frac{N^2}{E}.$$

Under this setting, the weighted Frobenius norm becomes the standard Frobenius norm, and the weighted similarity measure becomes the cut norm. Consider the induced graphon signals $W_{\boldsymbol{A}}$ and $W_{\boldsymbol{C}^*}$. Note that under the standard Frobenius norm, $W_{\boldsymbol{C}^*}$ satisfies Equation (21), as $W_{\boldsymbol{C}^*}$ is the SVD of $W_{\boldsymbol{A}}$, and is also the best Frobenius norm approximator. Hence, the bound in Equation (23) is satisfied in probability $1 - 1/R$, and becomes

$$\|W_{\boldsymbol{A}} - W_{\boldsymbol{C}^*}\|_{\square} \leq \|W_{\boldsymbol{A}}\|_{\mathrm{F}} \sqrt{\frac{R}{K}}.$$

We immediately get in probability $1 - 1/R$

$$\|\boldsymbol{A} - \boldsymbol{C}^*\|_{\square} < \|\boldsymbol{A}\|_{\mathrm{F}} \sqrt{\frac{R}{K}},$$

as required. □

The initialization is based on Corollary G.1 restricted to induced graphon-signals with a densifying cut similarity with a constant weight $q_{i,j} = N^2/E$. Consider $\boldsymbol{C} = \boldsymbol{U}_{K/4} \boldsymbol{\Sigma}_{K/4} \boldsymbol{V}_{K/4}^{\mathrm{T}}$, the rank $K/4$ singular value decomposition of $\boldsymbol{A}$. We get

$$\|\boldsymbol{A} - \boldsymbol{C}\|_{\square;N,E} < \frac{N}{\sqrt{E}} \sqrt{\frac{4R}{K}}. \tag{31}$$

We note that while the lemma guarantees a good initialization for the graph, the goodness is not measured in terms of the graph-signal densifying similarity, but rather with respect to cut metric. Still, we find that in practice this initialization improves performance significantly.

It is important to note that the method described in the previous section relies on computing the singular value decomposition (SVD) of the graph's adjacency matrix. Traditional algorithms for SVD require the entire adjacency matrix to be loaded into memory, which can be computationally expensive for large graphs. Similar to the approach proposed for computing the IBG in Appendix H, we aim to allow SVD computation without loading the entire edge set into memory.

In the following subsection, we introduce a Monte Carlo algorithm for computing the SVD of a matrix by processing only a random small fraction of its rows, significantly reducing memory requirements.

G.2 Monte Carlo SVD algorithm

For very large graphs, it is impossible to load the whole graph to GPU memory and estimate the SVD with standard algorithms. Instead, we propose here a randomized SVD algorithm which loads the matrix to GPU memory in small enough chunks.

The following theorem relates eigendecomposition of symmetric matrices to SVD [45, Lecture 31].

Theorem G.2. *Let $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ with singular values $\sigma_1 \geq \ldots \geq \sigma_N$, left singular vectors $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n$, and right singular vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_N$. Consider the* augmented matrix

$$\boldsymbol{B} = \begin{pmatrix} \boldsymbol{0} & \boldsymbol{A}^\top \\ \boldsymbol{A} & \boldsymbol{0} \end{pmatrix}.$$

Then, the eigenvalues of $\boldsymbol{B}$ are exactly $\sigma_1, \ldots, \sigma_N, -\sigma_N, \ldots, -\sigma_1$, and the eigenvectors are all vectors of the form

$$\frac{1}{\sqrt{2}} \begin{pmatrix} \boldsymbol{v}_i \\ \pm \boldsymbol{u}_i \end{pmatrix}, \quad i = 1, \ldots, N. \tag{32}$$

Theorem G.2 is the standard way to convert eigendecomposition algorithms of symmetric matrices to SVD of general matrices. Namely, one applies the eigendecomposition algorithm to the symmetric augmented matrix $\boldsymbol{B}$, and reads the singular values and vectors from $\sigma_1, \ldots, \sigma_N, -\sigma_N, \ldots, -\sigma_1$ and from (32).

We hence start with a basic randomized eigendecomposition algorithm for symmetric matrices, based on the simultaneous iteration (also called the block power method) [45, Part V].

Given a symmetric matrix $\boldsymbol{C}$, the standard *simultaneous iteration* (Algorithm 1) is an algorithm for finding the leading (largest in their absolute values) M eigenvector-eigenvalue pairs of $\boldsymbol{C}$. We next replace the full matrix product $\boldsymbol{CQ}$ by a Monte Carlo method, initially proposed in [46]. Consider a matrix $\boldsymbol{C} \in \mathbb{R}^{N\times N}$ with columns $\boldsymbol{c}_1, \ldots, \boldsymbol{c}_N \in \mathbb{R}^N$ and a column vector $\boldsymbol{v} = (v_1, \ldots, v_N) \in \mathbb{R}^N$. Let $m_1, \ldots, m_J$ be chosen independently uniformly at random from $[N]$. Let us denote $\boldsymbol{m} = (m_1, \ldots, m_J)$ and

$$[\boldsymbol{Cv}]_{\boldsymbol{m}} = \sum_{j=1}^{J} v_{m_j} \boldsymbol{c}_{m_j}.$$

One can show that in high probability

$$[\boldsymbol{Cv}]_{\boldsymbol{m}} \approx \boldsymbol{Cv},$$

where the expected square error satisfies

$$\mathbb{E}\|[\boldsymbol{Cv}]_{\boldsymbol{m}} - \boldsymbol{Cv}\|_2^2 = O(1/J).$$

The advantage in the computation $[\boldsymbol{Cv}]_{\boldsymbol{m}}$ is that it reduces the computational complexity of matrix-vector product from $O(N^2)$ to $O(JN)$.

We hence consider a *Monte Carlo simultaneous iteration algorithm*, which is identical to Algorithm 1 with the exception that the matrix-vector product $\boldsymbol{Z} = \boldsymbol{CQ}$ is approximated by $[\boldsymbol{CQ}]_{\boldsymbol{m}}$.

Lastly, we would like to use the Monte Carlo simultaneous iteration algorithm for estimating the SVD of a matrix $\boldsymbol{A}$ via Theorem G.2. For that, we require an additional consideration. Note that the simultaneous iteration finds the leading eigenvalues and eigenvectors (up to sign) in case they are distinct in their absolute value. In case $\lambda_i \neq \lambda_j$ but $|\lambda_i| = |\lambda_j|$, the simultaneous iteration would find a vector in the span of the eigenspaces corresponding to λ_i and λ_j. Now, note that Theorem G.2 builds the SVD of $\boldsymbol{A}$ via the eigendecomposition of $\boldsymbol{B}$, and every value λ_i of $\boldsymbol{B}$ is repeated twice, with positive and negative sign (in case the singular values are not repeated), and Algorithm 2 finds vectors only in the span of the two eigenspaces corresponding to $\pm\lambda_i$. This is not an issue in the exact algorithm, as the singular vectors can be read off (32) regardless of the mixing between eigenspaces. However, in the Monte Carlo simultaneous iteration algorithm, the algorithm separates the two dimensional spaces of $\boldsymbol{B}$ to one-dimensional spaces arbitrarily due to the inexact Monte Carlo matrix product. Moreover, the split of the space to two one dimensional spaces arbitrarily changes between iterations. Hence, when naively implemented, the SVD algorithm based on Theorem G.2 fails to converge.

Instead, we apply the Monte Carlo eigendecomposition algorithm on $\boldsymbol{B} + \lambda_1 \mathbf{I}$, where λ_1 is the largest eigenvalue of $\boldsymbol{B}$, computed by a Monte Carlo power iteration on $\boldsymbol{B}$. Since the addition of $\lambda_1 \mathbf{I}$ shifts the spectrum of $\boldsymbol{B}$ by λ_1, all eigenvalues of $\boldsymbol{B}$ become non-negative and distinct (under the assumption that the singular values of $\boldsymbol{A}$ are distinct). We summarize in Algorithm 2 the resulting method.

Algorithm 1 Simultaneous Iteration for Eigendecomposition of Symmetric Matrix

Input: Matrix $\boldsymbol{C} \in \mathbb{R}^{N\times N}$, number of leading eigenvalues to be computed M, number of iterations J.
Initialize $\boldsymbol{Q} \in \mathbb{R}^{N\times M}$ randomly
for $i = 1$ **to** J **do**
  $\boldsymbol{Z} = \boldsymbol{CQ}$
  $(\boldsymbol{Q}, \boldsymbol{R}) =$ Reduced-QR-Factorization$(\boldsymbol{Z})$
end for
Compute $\boldsymbol{\lambda} = (\lambda_1, \ldots, \lambda_M)$ with $\lambda_j = \boldsymbol{Q}[:, j]^\top \boldsymbol{CQ}[:, j]$
Output: Approximate eigenvalues $\lambda_1, \ldots, \lambda_M$ and corresponding columns $\boldsymbol{v_1}, \ldots, \boldsymbol{v_M}$ of $\boldsymbol{Q}$ as the approximate eigenvectors.

Algorithm 2 Monte Carlo Simultaneous Iteration for SVD decomposition of Non-Symmetric Matrix

Input: Matrix $\boldsymbol{C} \in \mathbb{R}^{N \times N}$, number of leading eigenvalues to be computed M, number of iterations J, sample ratio $0 < r \leq 1$.

{*Augment the non-symmetric matrix to be symmetric*}

Define $\boldsymbol{B} = \begin{pmatrix} \mathbf{0} & \boldsymbol{C}^\top \\ \boldsymbol{C} & \mathbf{0} \end{pmatrix}$.

{*Leading eigenvalue estimation*}

Initialize $\boldsymbol{Q}_1 \in \mathbb{R}^{2N \times 1}$ randomly
for $i = 1$ **to** J **do**
  Generate $2N \cdot r$ samples $\boldsymbol{n}$ of indices from $[2N]$ with repetitions.
  $\tilde{\boldsymbol{B}} = \boldsymbol{B}[:, \boldsymbol{n}]$
  $\tilde{\boldsymbol{Q}}_1 = \boldsymbol{Q}_1[\boldsymbol{n}, 1]$
  $\boldsymbol{Z} = \tilde{\boldsymbol{B}}\tilde{\boldsymbol{Q}}_1$
  $(\boldsymbol{Q}_1, \boldsymbol{R}) = \text{Reduced-QR-Factorization}(\boldsymbol{Z})$
end for
$\lambda_1 = \boldsymbol{Q}_1^\top \boldsymbol{B} \boldsymbol{Q}_1$

{*Shift the spectrum of B*}

$\hat{\boldsymbol{B}} = \boldsymbol{B} + |\lambda_1| \, \boldsymbol{I}_{2N}$

{*Randomized simultaneous iteration on $\hat{B}$*}

Initialize $\boldsymbol{Q} \in \mathbb{R}^{2N \times M}$ randomly
for $i = 1$ **to** J **do**
  Sample vector $\boldsymbol{n}$ of $2N \cdot r$ indices from $[2N]$ with repetitions.
  $\tilde{\boldsymbol{B}} = \hat{\boldsymbol{B}}[:, \boldsymbol{n}]$
  $\tilde{\boldsymbol{Q}} = \boldsymbol{Q}[\boldsymbol{n}, :]$
  $\boldsymbol{Z} = \tilde{\boldsymbol{B}}\tilde{\boldsymbol{Q}}$
  $(\boldsymbol{Q}, \boldsymbol{R}) = \text{Reduced-QR-Factorization}(\boldsymbol{Z})$
end for

{*Generate the output*}

Sample a vector $\boldsymbol{n}$ of $2N \cdot r$ indices from $[2N]$ with repetitions.
$\tilde{\boldsymbol{B}} = \boldsymbol{B}[:, \boldsymbol{n}]$
$\tilde{\boldsymbol{Q}} = \boldsymbol{Q}[\boldsymbol{n}, :]$
Compute $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_M)$ with $\sigma_j = \boldsymbol{Q}[:, j]^\top \tilde{\boldsymbol{B}} \tilde{\boldsymbol{Q}}[:, j]$
$\boldsymbol{Q}_v = \boldsymbol{Q}[1 : N, :]$
$\boldsymbol{Q}_u = \boldsymbol{Q}[N + 1 : 2N, :]$

Output: Approximate singular values $\sigma_1, \ldots, \sigma_m$ and corresponding columns $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m$ and $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m$ of $\boldsymbol{Q}_u$ and $\boldsymbol{Q}_v$ as the approximate left and right singular vectors.

H Learning IBG with subgraph SGD

For message-passing neural networks, processing large graphs becomes challenging when the number of edges E exceeds the capacity of GPU memory. For this reason, processing IBGs with neural networks, which take $\mathcal{O}(N)$ operations, is advantageous. However, one still has to fit the IBG to the graph as a preprocessing step, which takes $\mathcal{O}(E)$ operations and memory complexity. To address this, we propose two sampling-based optimization methods for the IBG. The first method performs node sampling, extending the SGD approach of [11]. The second samples individual entries of the adjacency matrix, which we refer to as diodes. We term these methods node-sampling SGD and diode-sampling SGD, accordingly.

H.1 Construction of Node sampling SGD

In a standard GD procedure, all edges of the graph are loaded into memory. Instead, in our node-sampling SGD procedure, a set of $M \ll N$ nodes is sampled uniformly with replacement from $[N]$. Denote these nodes by $\boldsymbol{n} := (n_m)_{m=1}^M$. We then perform the gradient step using the gradients calculated solely using these sampled nodes. More specifically, by denoting $\boldsymbol{A}^{(\boldsymbol{n})} \in \mathbb{R}^{M\times M}$ the sampled subgraph with entries $a_{i,j}^{(\boldsymbol{n})} = a_{n_i,n_j}$, $\boldsymbol{X}^{(\boldsymbol{n})} \in \mathbb{R}^{M\times K}$ the sampled sub-signal with entries $\boldsymbol{x}_i^{(\boldsymbol{n})} = \boldsymbol{x}_{n_i}$, and $\boldsymbol{U}^{(\boldsymbol{n})}, \boldsymbol{V}^{(\boldsymbol{n})} \in [0,1]^{M\times K}$ as the sampled community target and source affiliation matrices with entries $u_{i,j}^{(\boldsymbol{n})} = u_{n_i,j}$ and $v_{i,j}^{(\boldsymbol{n})} = v_{n_i,j}$. The loss over the sampled nodes becomes:

$$L^{(M)}(\boldsymbol{U}^{(\boldsymbol{n})}, \boldsymbol{V}^{(\boldsymbol{n})}, \boldsymbol{r}, \boldsymbol{F}, \boldsymbol{B}) = \frac{\alpha(1+\Gamma)\mu}{M^2}\sum_{i,j=1}^{M}\sum_{k=1}^{K}(u_{n_i,k}r_k v_{n_j,k} - a_{n_i,n_j})^2 q_{n_i,n_j} + \frac{\beta}{MD}\sum_{i=1}^{M}\sum_{d=1}^{D}\sum_{k=1}^{K}(u_{n_i,k}f_{k,d} + v_{n_i,k}b_{k,d} - x_{n_i,d})^2,$$

H.2 Construction of Diode sampling SGD

We sample a set of $M \ll N^2$ diodes $\mathcal{D} = (\boldsymbol{s}, \boldsymbol{t}) = (s_m, t_m)_{m=1}^M$ independently from a distribution we define over all diodes of the graph. The probability of sampling diode (i,j) is $\frac{q_{ij}}{(1+\Gamma)E}$, and the loss over the sampled nodes is

$$L^{(M)}(\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{r}) = \frac{\alpha(1+\Gamma)}{M}\sum_{m=1}^{M}\sum_{k=1}^{K}(u_{s_m,k}r_k v_{t_m,k} - a_{s_m,t_m})^2.$$

We then perform the gradient step using the gradients calculated solely using these sampled diodes.

We note that for diode-sampling SGD, we define the loss only over the graph part. To perform SGD over the signal part of the loss, the method is identical to node-sampling SGD.

H.3 Theoretical analysis of diode-sampling SGD

In the following section, we prove that the gradients calculated using diode-sampling SGD with respect to the sampled diodes approximate those calculated over the full graph using the standard loss in (8). Throughout the section we denote $\mu = 1/\sum q_{i,j}$ and refer only to the sample loss of diode-sampling SGD.

Proposition H.1. *Let $0 < p < 1$. Consider the Frobenius loss weighted by $\boldsymbol{Q}$, α and β. If we restrict all entries of $\boldsymbol{C}$, $\boldsymbol{U}$, $\boldsymbol{V}$, and $\boldsymbol{r}$ to be in $[-1,1]$, then in probability at least $1-p$, for every $k \in [K]$ and $m \in [M]$*

$$\left|\nabla_{u_{n_m,k}} L - \nabla_{u_{n_m,k}} L^{(M)}\right| \le 2\alpha(1+\Gamma)\mu\sqrt{\frac{2\log(1/p) + 2\log(K) + 2\log(6)}{M}},$$
$$\left|\nabla_{v_{n_t,k}} L - \nabla_{v_{n_m,k}} L^{(M)}\right| \le 2\alpha(1+\Gamma)\mu\sqrt{\frac{2\log(1/p) + 2\log(K) + 2\log(6)}{M}},$$
$$\left|\nabla_{r_k} L - \nabla_{r_k} L^{(M)}\right| \le 2\alpha(1+\Gamma)\mu\sqrt{\frac{2\log(1/p) + 2\log(K) + 2\log(6)}{M}}.$$

Proposition H.1 provide a probabilistic bound on the difference between the gradients computed on the full graph and those calculated using subgraph SGD. Notably, it shows that the gradients of the IBG parameters calculated with SGD closely approximate those calculated with standard GD.

We prove Proposition H.1 by comparing the gradients of the full loss L with those of the sampled loss $L^{(M)}$.

Consider the graph part of the IBG loss:

$$L(\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{r}) = \alpha(1+\Gamma)\mu \sum_{i,j=1}^{N} \sum_{k=1}^{K} (u_{i,k} r_k v_{j,k} - a_{i,j})^2 q_{i,j}$$

Next, we calculate the gradients of $\boldsymbol{C} = \boldsymbol{U}\,\mathrm{diag}(\boldsymbol{r})\boldsymbol{V}^\top$ with respect to $\boldsymbol{U}$, $\boldsymbol{V}$, and $\boldsymbol{r}$ in coordinates. We have

$$\nabla_{r_k} c_{i,j} = u_{i,k} v_{j,k},$$
$$\nabla_{u_{t,k}} c_{i,j} = r_k \delta_{i-t} v_{j,k},$$

and

$$\nabla_{v_{t,k}} c_{i,j} = r_k \delta_{j-t} u_{i,k},$$

where δ_i is 1 if $i = 0$ and zero otherwise. Hence, the gradients of the loss are

$$\nabla_{r_k} L = 2\alpha(1+\Gamma)\mu \sum_{i,j=1}^{N} (c_{i,j} - a_{i,j}) q_{i,j} u_{i,k} v_{j,k},$$

$$\nabla_{u_{n,k}} L = 2\alpha(1+\Gamma)\mu \sum_{j=1}^{N} (c_{n,j} - a_{n,j}) q_{n,j} r_k v_{j,k},$$

and

$$\nabla_{v_{n,k}} L = 2\alpha(1+\Gamma)\mu \sum_{j=1}^{N} (c_{j,n} - a_{j,n}) q_{j,n} r_k u_{j,k}.$$

Similarly

$$\nabla_{r_k} L^{(M)} = \frac{2\alpha(1+\Gamma)}{M} \sum_{j=1}^{M} (c_{s_j,t_j} - a_{s_j,t_j}) u_{s_j,k} v_{t_j,k},$$

$$\nabla_{u_{n_m,k}} L^{(M)} = \frac{2\alpha(1+\Gamma)\mu}{M} \sum_{j=1}^{M} (c_{n_m,t_j} - a_{n_m,t_j}) r_k v_{t_j,k},$$

and

$$\nabla_{v_{n_m,k}} L^{(M)} = \frac{2\alpha(1+\Gamma)\mu}{M} \sum_{j=1}^{M} (c_{s_j,n_m} - a_{s_j,n_m}) r_k u_{s_j,k},$$

The following convergence analysis is based on Hoeffding's inequality, and a supporting Monte Carlo approximation lemma.

Theorem H.2 (Hoeffding's Inequality)**.** *Let $Y_1, \ldots, Y_M$ be independent random variables such that $a \leq Y_m \leq b$ almost surely. Then, for every $k > 0$,*

$$\mathbb{P}\Big(\Big|\frac{1}{M}\sum_{m=1}^{M}(Y_m - \mathbb{E}[Y_m])\Big| \geq k\Big) \leq 2\exp\Big(-\frac{2k^2 M}{(b-a)^2}\Big).$$

We now use Hoeffding's inequality to derive a standard Monte Carlo approximation error bound.

Lemma H.3. *Let $\{i_m\}_{m=1}^M$ be uniform i.i.d in $[N]$. Let $\boldsymbol{v} \in \mathbb{R}^N$ be a vector with entries v_n in the set $[-1, 1]$. Then, for every $0 < p < 1$, in probability at least $1 - p$*

$$\left|\frac{1}{M}\sum_{m=1}^{M} v_{i_m} - \frac{1}{N}\sum_{n=1}^{N} v_n\right| \leq \sqrt{\frac{2\log(1/p) + 2\log(2)}{M}}.$$

Proof. This is a direct result of Hoeffding's Inequality on the i.i.d. variables $\{v_{i_m}\}_{m=1}^M$. □

We use Lemma H.3 on the i.i.d samples $\boldsymbol{m}$. We first show:

$$\mathbb{E}[\nabla_{r_k} L^{(M)}] = \nabla_{r_k} L,$$

$$\mathbb{E}[\nabla_{u_{n_m,k}} L^{(M)}] = \nabla_{u_{n_m,k}} L,$$

and

$$\mathbb{E}[\nabla_{v_{n_m,k}} L^{(M)}] = \nabla_{v_{n_m,k}} L.$$

Indeed we have

$$\mathbb{E}[\nabla_{r_k} L^{(M)}] = \frac{2\alpha(1+\Gamma)}{M} \sum_{j=1}^{M} \mathbb{E}[(c_{s_j,t_j} - a_{s_j,t_j}) u_{s_j,k} v_{t_j,k}] =$$

$$= \frac{2\alpha(1+\Gamma)}{M} \sum_{j=1}^{M} \mathbb{E}[(c_{s_1,t_1} - a_{s_1,t_1}) u_{s_1,k} v_{t_1,k}] =$$

$$= 2\alpha(1+\Gamma) \sum_{ij=1}^{N} P(s=i, t=j)(c_{i,j} - a_{i,j}) u_{i,k} v_{j,k} =$$

$$= 2\alpha(1+\Gamma)\mu \sum_{ij=1}^{N} (c_{i,j} - a_{i,j}) q_{ij} u_{i,k} v_{j,k},$$

$$\mathbb{E}[\nabla_{u_{n_m,k}} L^{(M)}] = \frac{2\alpha(1+\Gamma)}{M} \sum_{j=1}^{M} \mathbb{E}[(c_{n_m,t_j} - a_{n_m,t_j}) r_k v_{t_j,k}] =$$

$$= \frac{2\alpha(1+\Gamma)}{M} \sum_{j=1}^{M} \mathbb{E}[(c_{n_m,t_1} - a_{n_m,t_1}) r_k v_{t_1,k}] =$$

$$= 2\alpha(1+\Gamma)\mu \sum_{j=1}^{N} (c_{n_m,j} - a_{n_m,j}) q_{n_m j} r_k v_{j,k},$$

and

$$\mathbb{E}[\nabla_{v_{n_m,k}} L^{(M)}] = \frac{2\alpha(1+\Gamma)}{M} \sum_{j=1}^{M} \mathbb{E}[(c_{n_m,t_j} - a_{n_m,t_j}) r_k v_{t_j,k}] =$$

$$= \frac{2\alpha(1+\Gamma)}{M} \sum_{j=1}^{M} \mathbb{E}[(c_{n_m,t_1} - a_{n_m,t_1}) r_k v_{t_1,k}] =$$

$$= 2\alpha(1+\Gamma)\mu \sum_{j=1}^{N} (c_{n_m,j} - a_{n_m,j}) q_{n_m j} r_k v_{j,k}.$$

This shows that the expected value of the sampled loss gradients is equal to the standard loss's gradients, which meets the conditions of Lemma H.3. We therefore use the lemma to obtain a probabilistic bound on the difference between the approximated gradients and the full gradients.

Specifically, for any $0 < p_1 < 1$, for every $k \in [K]$ there is an event $\mathcal{A}_k$ of probability at least $1 - p_1$ such that

$$\left| \nabla_{r_k} L - \nabla_{r_k} L^{(M)} \right| \leq 2\alpha(a+\Gamma)\mu \sqrt{\frac{2\log(1/p_1) + 2\log(2)}{M}}.$$

For any $0 < p_2 < 1$, for every $k \in [K]$ there is an event $\mathcal{U}_k$ of probability at least $1 - p_2$ such that for every $n \in [N]$

$$\left| \nabla_{u_{n,k}} L - \nabla_{u_{n,k}} L^{(M)} \right| \leq 2\alpha(a+\Gamma)\mu \sqrt{\frac{2\log(1/p_2) + 2\log(2)}{M}}.$$

For any $0 < p_3 < 1$, for every $k \in [K]$ there is an event $\mathcal{V}_k$ of probability at least $1 - p_3$ such that for every $n \in [N]$

$$\left|\nabla_{v_{n,k}} L - \nabla_{v_{n,k}} L^{(M)}\right| \leq 2\alpha(a+\Gamma)\mu\sqrt{\frac{2\log(1/p_3) + 2\log(2)}{M}}.$$

Lastly, given $0 < p < 1$, choosing $p_1 = p_2 = p_3 = p/3K$ and intersecting all events for all coordinates gives in the event $\mathcal{E}$ of probability at least $1 - p$

$$\left|\nabla_{r_k} L - \nabla_{r_k} L^{(M)}\right| \leq 2\alpha(a+\Gamma)\mu\sqrt{\frac{2\log(1/p) + 2\log(K) + 2\log(6)}{M}},$$

$$\left|\nabla_{u_{n,k}} L - \nabla_{u_{n,k}} L^{(M)}\right| \leq 2\alpha(a+\Gamma)\mu\sqrt{\frac{2\log(1/p) + 2\log(K) + 2\log(6)}{M}},$$

and

$$\left|\nabla_{v_{n,k}} L - \nabla_{v_{n,k}} L^{(M)}\right| \leq 2\alpha(a+\Gamma)\mu\sqrt{\frac{2\log(1/p) + 2\log(K) + 2\log(6)}{M}}.$$

proving Proposition H.1

H.4 Theoretical analysis of node sampling SGD

In this section, we provide a similar result to the one in Appendix H.3, proving that the gradients calculated using node-sampling SGD with respect to the sampled nodes approximate those calculated over the full graph using the standard loss in (8). Throughout the section we use the loss of node-sampling SGD.

Proposition H.4. *Let $0 < p < 1$. Consider the Frobenius loss weighted by $\boldsymbol{Q}$, α and β. If we restrict all entries of $\boldsymbol{C}$, $\boldsymbol{P}$, $\boldsymbol{U}$, $\boldsymbol{V}$, $\boldsymbol{r}$, $\boldsymbol{F}$ and $\boldsymbol{B}$ to be in $[-1, 1]$, then in probability at least $1 - p$, for every $k \in [K]$, $d \in [D]$ and $m \in [M]$*

$$\left|\nabla_{u_{n_m,k}} L - \frac{M}{N}\nabla_{u_{n_m,k}} L^{(M)}\right| \leq 2\alpha(1+\Gamma)\mu N\sqrt{\frac{2\log(1/p) + 2\log(2N) + 2\log(K) + 2\log(10)}{M}},$$

$$\left|\nabla_{v_{n_t,k}} L - \frac{M}{N}\nabla_{v_{n_m,k}} L^{(M)}\right| \leq 2\alpha(1+\Gamma)\mu N\sqrt{\frac{2\log(1/p) + 2\log(2N) + 2\log(K) + 2\log(10)}{M}},$$

$$\left|\nabla_{r_k} L - \nabla_{r_k} L^{(M)}\right| \leq 4\alpha(1+\Gamma)\mu N^2\sqrt{\frac{2\log(1/p) + 2\log(N) + 2\log(K) + 2\log(10)}{M}},$$

$$\left|\nabla_{f_{k,d}} L - \nabla_{f_{k,d}} L^{(M)}\right| \leq \frac{4\beta}{D}\sqrt{\frac{2\log(1/p) + 2\log(K) + 2\log(D) + 2\log(10)}{M}},$$

$$\left|\nabla_{b_{k,d}} L - \nabla_{b_{k,d}} L^{(M)}\right| \leq \frac{4\beta}{D}\sqrt{\frac{2\log(1/p) + 2\log(K) + 2\log(D) + 2\log(10)}{M}}.$$

The proof of Proposition H.4 is similar to that of Proposition H.1. Define the graph part and the signal part of the IBG loss

$$L_1(\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{r}) = \frac{\alpha(1+\Gamma)\mu}{N^2}\sum_{i,j=1}^{N}\sum_{k=1}^{K}(u_{i,k} r_k v_{j,k} - a_{i,j})^2 q_{i,j}$$

$$L_2(\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{F}, \boldsymbol{B}) = \frac{\beta}{ND}\sum_{i=1}^{N}\sum_{d=1}^{D}\sum_{k=1}^{K}(u_{i,k} f_{k,d} + v_{i,k} b_{k,d} - x_{i,d})^2.$$

The IBG loss in (8) is $L = N^2 L_1 + L_2$. We normalize and multiply L_1 by N^2 for reasons that will become clear later.

Similarly, we define the graph and signal parts of the subgraph SGD loss

$$L_1^{(M)}(\boldsymbol{U}^{(\boldsymbol{n})}, \boldsymbol{V}^{(\boldsymbol{n})}, \boldsymbol{r}) = \frac{\alpha(1+\Gamma)\mu}{M^2}\sum_{i,j=1}^{M}\sum_{k=1}^{K}(u_{n_i,k} r_k v_{n_j,k} - a_{n_i,n_j})^2 q_{n_i,n_j}$$

$$L_2^{(M)}(\boldsymbol{U}^{(\boldsymbol{n})}, \boldsymbol{V}^{(\boldsymbol{n})}, \boldsymbol{F}, \boldsymbol{B}) = \frac{\beta}{MD}\sum_{i=1}^{M}\sum_{d=1}^{D}\sum_{k=1}^{K}(u_{n_i,k}f_{k,d} + v_{n_i,k}b_{k,d} - x_{n_i,d})^2,$$

where $L^{(M)} = N^2 L_1^{(M)} + L_2^{(M)}$

Next, we extend the calculations of the gradients of $\boldsymbol{C} = \boldsymbol{U}\,\mathrm{diag}(\boldsymbol{r})\boldsymbol{V}^\top$ and $\boldsymbol{P} = \boldsymbol{U}\boldsymbol{F} + \boldsymbol{V}\boldsymbol{B}$ with respect to $\boldsymbol{U}$, $\boldsymbol{V}$, $\boldsymbol{r}$, $\boldsymbol{F}$ and $\boldsymbol{B}$ in coordinates. We have

$$\nabla_{f_{k,l}} p_{i,d} = u_{i,k}\delta_{l-d},$$
$$\nabla_{b_{k,l}} p_{i,d} = v_{i,k}\delta_{l-d},$$
$$\nabla_{u_{t,k}} p_{i,d} = f_{k,d}\delta_{i-t},$$

and

$$\nabla_{v_{t,k}} p_{i,d} = b_{k,d}\delta_{i-t}.$$

Hence,

$$\nabla_{r_k} L_1 = \frac{\alpha(1+\Gamma)\mu}{N^2}\sum_{i,j=1}^{N}(c_{i,j} - a_{i,j})q_{i,j}u_{i,k}v_{j,k},$$

$$\nabla_{u_{t,k}} L_1 = \frac{\alpha(1+\Gamma)\mu}{N^2}\sum_{j=1}^{N}(c_{t,j} - a_{t,j})q_{t,j}r_k v_{j,k} \quad \nabla_{u_{t,k}} L_2 = \frac{\beta}{ND}\sum_{d=1}^{D}(p_{t,d} - x_{t,d})f_{k,d},$$

$$\nabla_{v_{t,k}} L_1 = \frac{\alpha(1+\Gamma)\mu}{N^2}\sum_{j=1}^{N}(c_{t,j} - a_{t,j})q_{t,j}r_k u_{j,k} \quad \nabla_{v_{t,k}} L_2 = \frac{\beta}{ND}\sum_{d=1}^{D}(p_{t,d} - x_{t,d})b_{k,d},$$

$$\nabla_{f_{k,d}} L_2 = \frac{\beta}{ND}\sum_{i=1}^{N}(p_{i,d} - x_{i,d})u_{i,k},$$

and

$$\nabla_{b_{k,d}} L_2 = \frac{\beta}{ND}\sum_{i=1}^{N}(p_{i,d} - x_{i,d})v_{i,k}.$$

Similarly

$$\nabla_{r_k} L_1^{(M)} = \frac{\alpha(1+\Gamma)\mu}{M^2}\sum_{i,j=1}^{M}(c_{n_i,n_j} - a_{n_i,n_j})q_{n_i,n_j}u_{n_i,k}v_{n_j,k},$$

$$\nabla_{u_{n_m,k}} L_1^{(M)} = \frac{\alpha(1+\Gamma)\mu}{M^2}\sum_{j=1}^{M}(c_{n_m,n_j} - a_{n_m,n_j})q_{n_m,n_j}r_k v_{n_j,k}$$

$$\nabla_{u_{n_m,k}} L_2^{(M)} = \frac{\beta}{MD}\sum_{d=1}^{D}(p_{n_m,d} - x_{n_m,d})f_{k,d},$$

$$\nabla_{v_{n_m,k}} L_1^{(M)} = \frac{\alpha(1+\Gamma)\mu}{M^2}\sum_{j=1}^{M}(c_{n_m,n_j} - a_{n_m,n_j})q_{n_m,n_j}r_k u_{n_j,k}$$

$$\nabla_{v_{n_m,k}} L_2^{(M)} = \frac{\beta}{MD}\sum_{d=1}^{D}(p_{n_m,d} - x_{n_m,d})b_{k,d},$$

$$\nabla_{f_{k,d}} L_2^{(M)} = \frac{\beta}{MD}\sum_{i=1}^{M}(p_{n_i,d} - x_{n_i,d})u_{n_i,k},$$

and

$$\nabla_{b_{k,d}} L_2^{(M)} = \frac{\beta}{MD}\sum_{i=1}^{M}(p_{n_i,d} - x_{n_i,d})v_{n_i,k}.$$

The following Lemma provides an error bound between the sum of a 2D array of numbers and the sum of random points from the 2D array. We study the setting where one randomly (and independently)

samples only points in a 1D axis, and the 2D random samples consist of all pairs of these samples. This results in dependent 2D samples, but still, one can prove a Monte Carlo-type error bound in this situation. The Lemma is similar to Lemma E.3. in Finkelshtein et al. [11], generalized to non-symmetric matrices.

Lemma H.5. *Let $\{i_m\}_{m=1}^M$ be uniform i.i.d in $[N]$. Let $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ with $a_{i,j} \in [-1,1]$. Then, for every $0 < p < 1$, in probability more than $1-p$*

$$\left| \frac{1}{N^2}\sum_{j=1}^{N}\sum_{n=1}^{N} a_{j,n} - \frac{1}{M^2}\sum_{m=1}^{M}\sum_{l=1}^{M} a_{i_m,i_l} \right| \leq 2\sqrt{\frac{2\log(1/p)+2\log(2N)+2\log(2)}{M}}.$$

Proof. Let $0 < p < 1$. For each fixed $n \in [N]$, consider the independent random variables $Y_m^n = a_{i_m,n}$, with

$$\mathbb{E}(Y_m^n) = \frac{1}{N}\sum_{j=1}^{N} a_{j,n}$$

and $-1 \leq Y_m \leq 1$.
Similarly define the independent random variables $W_m^n = a_{n,i_m}$ with

$$\mathbb{E}(W_m^n) = \frac{1}{N}\sum_{j=1}^{N} a_{n,j}.$$

By Hoeffding's Inequality, for $k = \sqrt{\frac{2\log(1/p)+2\log(2N)+2\log(2)}{M}}$, we have

$$\left| \frac{1}{N}\sum_{j=1}^{N} a_{j,n} - \frac{1}{M}\sum_{m=1}^{M} a_{i_m,n} \right| \leq k$$

and

$$\left| \frac{1}{N}\sum_{j=1}^{N} a_{n,j} - \frac{1}{M}\sum_{m=1}^{M} a_{n,i_m} \right| \leq k$$

in the event $\mathcal{E}_n^Y$ and $\mathcal{E}_n^W$ of probability more than $1 - p/2N$. Intersecting the events $\{\mathcal{E}_n^Y\}_{n=1}^N$ and $\{\mathcal{E}_n^W\}_{n=1}^N$, we get $\forall n \in [N]$:

$$\left| \frac{1}{N}\sum_{j=1}^{N} a_{j,n} - \frac{1}{M}\sum_{m=1}^{M} a_{i_m,n} \right| \leq k$$

and

$$\left| \frac{1}{N}\sum_{j=1}^{N} a_{n,j} - \frac{1}{M}\sum_{m=1}^{M} a_{n,i_m} \right| \leq k$$

in the event $\mathcal{E} = \cap_n \mathcal{E}_n^Y \cap_n \mathcal{E}_n^W$ with probability at least $1-p$. The rows and columns of $\boldsymbol{A}$ are not independent, meaning the probability of their intersection is at least $1-p$ and by the triangle inequality, we also have in the event $\mathcal{E}$

$$\left| \frac{1}{NM}\sum_{l=1}^{M}\sum_{j=1}^{N} a_{i_l,j} - \frac{1}{M^2}\sum_{l=1}^{M}\sum_{m=1}^{M} a_{i_l,i_m} \right| \leq k,$$

and

$$\left| \frac{1}{N^2}\sum_{n=1}^{N}\sum_{j=1}^{N} a_{j,n} - \frac{1}{NM}\sum_{n=1}^{N}\sum_{m=1}^{M} a_{i_m,n} \right| \leq k.$$

Hence, by the triangle inequality,

$$\left|\frac{1}{N^2}\sum_{n=1}^{N}\sum_{j=1}^{N}a_{j,n}-\frac{1}{M^2}\sum_{l=1}^{M}\sum_{m=1}^{M}a_{i_m,i_l}\right|\leq 2k.$$

□

We now derive bounds on the approximation errors for the gradients of L. Note that the gradients of the SGD loss with respect to each element of the IBG are

$$\begin{aligned}\nabla_{u_{n_m,k}}L^{(M)}&=N^2\nabla_{u_{n_m,k}}L^{(M)}+\nabla_{u_{n_m,k}}L_2^{(M)}\\ \nabla_{v_{n_m,k}}L^{(M)}&=N^2\nabla_{v_{n_m,k}}L_1^{(M)}+\nabla_{v_{n_m,k}}L_2^{(M)}\\ \nabla_{r_k}L^{(M)}&=N^2\nabla_{r_k}L_1^{(M)}\\ \nabla_{b_{k,m}}L^{(M)}&=\nabla_{b_{k,m}}L_2^{(M)}\\ \nabla_{f_{k,m}}L^{(M)}&=\nabla_{f_{k,m}}L_2^{(M)}\end{aligned}$$

We use Lemmas H.3 and H.5. These results are applicable in our setting because all entries of the relevant matrices and vectors, $\boldsymbol{A}$, $\boldsymbol{X}$, $\boldsymbol{C}$, $\boldsymbol{P}$, $\boldsymbol{U}$, $\boldsymbol{V}$, $\boldsymbol{r}$, $\boldsymbol{F}$, and $\boldsymbol{B}$ are bounded in $[-1,1]$.

Specifically, for any $0<p_1<1$, for every $k\in[K]$ there is an event $\mathcal{A}_k$ of probability at least $1-p_1$ such that

$$\left|\nabla_{r_k}L-\nabla_{r_k}L^{(M)}\right|\leq 4\alpha(1+\Gamma)\mu N^2\sqrt{\frac{2\log(1/p_1)+2\log(2N)+2\log(2)}{M}}.$$

Moreover, for every $k\in[K]$ and $l\in[D]$, and every $0<p_2<1$ there is an event $\mathcal{C}_{k,j}$ of probability at least $1-p_2$ such that

$$\left|\nabla_{f_{k,m}}L-\nabla_{f_{k,m}}L^{(M)}\right|\leq \frac{4\beta}{D}\sqrt{\frac{2\log(1/p_2)+2\log(2)}{M}}.$$

Similarly, for every $0<p_3<1$ there is an event $\mathcal{D}_{k,m}$ of probability at least $1-p_3$ such that

$$\left|\nabla_{b_{k,m}}L-\nabla_{b_{k,m}}L^{(M)}\right|\leq \frac{4\beta}{D}\sqrt{\frac{2\log(1/p_3)+2\log(2)}{M}}$$

For the approximation analysis of $\nabla_{u_{n_m,l}}L$ and $\nabla_{v_{n_m,l}}L$, note that the index n_i is random, so we derive a uniform convergence analysis for all possible values of n_i. For that, for every $n\in[N]$ and $k\in[K]$, define the vectors

$$\widetilde{\nabla_{u_{n,k}}L_1^{(M)}}=\frac{\alpha(1+\Gamma)\mu}{M^2}\sum_{j=1}^{M}(c_{n,n_j}-a_{n,n_j})q_{n,n_j}r_kv_{n_j,k}$$

$$\widetilde{\nabla_{u_{n,k}}L_2^{(M)}}=\frac{\beta}{MD}\sum_{d=1}^{D}(p_{n,d}-x_{n,d})f_{k,d},$$

and

$$\widetilde{\nabla_{v_{n,k}}L_1^{(M)}}=\frac{\alpha(1+\Gamma)\mu}{M^2}\sum_{j=1}^{M}(c_{n,n_j}-a_{n,n_j})q_{n,n_j}r_ku_{n_j,k}$$

$$\widetilde{\nabla_{v_{n,k}}L_2^{(M)}}=\frac{\beta}{MD}\sum_{d=1}^{D}(p_{n,d}-x_{n,d})b_{k,d}.$$

Note that $\widetilde{\nabla_{u_{n,k}}L^{(M)}}$ and $\widetilde{\nabla_{v_{n,k}}L^{(M)}}$ are not gradients of $L^{(M)}$ (since if n is not a sample from $\{n_i\}$ the gradient must be zero), but are denoted with ∇ for their structural similarity to $\nabla_{u_{n_m,k}}L^{(M)}$ and $\nabla_{v_{n_m,k}}L^{(M)}$. However, we get for every $m\in[M]$

$$\nabla_{u_{n_m,k}}L_2=\widetilde{\nabla_{u_{n_m,k}}L_2^{(M)}}$$

and

$$\nabla_{v_{n_m,k}} L_2 = \widetilde{\nabla_{v_{n_m,k}} L_2^{(M)}}$$

Hence, for every $m \in [M]$, we have

$$\nabla_{u_{n,k}} L^{(M)} = N^2 \widetilde{\nabla_{u_{n,k}} L_1^{(M)}}$$

and

$$\nabla_{v_{n,k}} L^{(M)} = N^2 \widetilde{\nabla_{v_{n,k}} L_1^{(M)}}.$$

Let $0 < p_4 < 1$. By Lemma H.3, for every $k \in [K]$ there is an event $\mathcal{U}_k$ of probability at least $1 - p_4$ such that for every $n \in [N]$

$$\left| \nabla_{u_{n,k}} L_1 - \frac{M}{N} \widetilde{\nabla_{u_{n,k}} L_1^{(M)}} \right| \leq \frac{\alpha(1+\Gamma)\mu}{N} \sqrt{\frac{2\log(1/p_4) + 2\log(N) + 2\log(2)}{M}},$$

Similarly, for $0 < p_5 < 1$ for every $k \in [K]$ there is an event $\mathcal{V}_k$ of probability at least $1 - p_5$ such that for every $n \in [N]$

$$\left| \nabla_{v_{n,k}} L_1 - \frac{M}{N} \widetilde{\nabla_{v_{n,k}} L_1^{(M)}} \right| \leq \frac{\alpha(1+\Gamma)\mu}{N} \sqrt{\frac{2\log(1/p_5) + 2\log(N) + 2\log(2)}{M}},$$

This means that in the event $\mathcal{U}_k$, for every $m \in [M]$ we have

$$\left| \nabla_{u_{n_m,k}} L - \frac{M}{N} \nabla_{u_{n_m,k}} L^{(M)} \right| \leq \alpha(1+\Gamma)\mu N \sqrt{\frac{2\log(1/p_4) + 2\log(N) + 2\log(2)}{M}},$$

and in the event $\mathcal{V}_k$, for every $m \in [M]$ we have

$$\left| \nabla_{v_{n_m,k}} L - \frac{M}{N} \nabla_{v_{n_m,k}} L^{(M)} \right| \leq \alpha(1+\Gamma)\mu N \sqrt{\frac{2\log(1/p_5) + 2\log(N) + 2\log(2)}{M}},$$

Lastly, given $0 < p < 1$, choosing $p_1 = p_4 = p_5 = p/5K$ and $p_2 = p_3 = p/5KD$ and intersecting all events for all coordinates gives in the event $\mathcal{E}$ of probability at least $1 - p$

$$\left| \nabla_{r_k} L - \nabla_{r_k} L^{(M)} \right| \leq 4\alpha(1+\Gamma)\mu N^2 \sqrt{\frac{2\log(1/p) + 2\log(N) + 2\log(K) + 2\log(10)}{M}}$$

$$\left| \nabla_{f_{k,m}} L - \nabla_{f_{k,m}} L^{(M)} \right| \leq \frac{2\beta}{D} \sqrt{\frac{2\log(1/p) + 2\log(K) + 2\log(D) + 2\log(10)}{M}},$$

$$\left| \nabla_{b_{k,m}} L - \nabla_{b_{k,m}} L^{(M)} \right| \leq \frac{2\beta}{D} \sqrt{\frac{2\log(1/p) + 2\log(K) + 2\log(D) + 2\log(10)}{M}},$$

$$\left| \nabla_{u_{n_m,k}} L - \frac{M}{N} \nabla_{u_{n_m,k}} L^{(M)} \right| \leq \alpha(1+\Gamma)\mu N \sqrt{\frac{2\log(1/p) + 2\log(N) + 2\log(K) + 2\log(10)}{M}},$$

and

$$\left| \nabla_{v_{n_m,k}} L - \frac{M}{N} \nabla_{v_{n_m,k}} L^{(M)} \right| \leq \alpha(1+\Gamma)\mu N \sqrt{\frac{2\log(1/p) + 2\log(N) + 2\log(K) + 2\log(10)}{M}},$$

thus proving Proposition H.4.

H.5 Comparison of Node sampling and Diode sampling

Propositions H.1 and H.4 both provide a probabilistic bound on the difference between the gradients calculated using subgraph SGD and the standard loss (8) using the full graph. Despite this similarity, these approces differ in their sampling schemes and computational tradeoffs.

For Proposition H.4, unlike Proposition H.1, only a subset of entries from $\boldsymbol{U}$ and $\boldsymbol{V}$ are involved in the loss computation per step. This introduces a time–memory tradeoff: since only a subset of $\boldsymbol{U}$ and $\boldsymbol{V}$ is updated in each step, the number of iterations required grows by a factor of N/M in expectation,

and the memory consumption reduces by M/N, where M denotes the number of sampled nodes. This mild growth in runtime is acceptable, as the number of iterations only grows by a linear scale, while each iteration becomes faster to complete due to the reduced size of the graph.

For Proposition H.1 we sample the diodes of the graph directly. Therefore potentially all nodes of the graph are involved in the calculation of the loss. Using this method, the time and memory complexity of calculating the loss becomes $\mathcal{O}(MK)$, where M is the number of sampled edges.

Notably, the bounds in Proposition H.1 are independent of the graph size, while the bounds in Proposition H.4 improve as the graph becomes more dense. However, in practice, we see that both diode sampling SGD and node sampling SGD provide good IBG approximations that work well for IBG-NN.

I Knowledge Graphs

I.1 Basic definitions

Knowledge graph signals. We consider knowledge graphs $G = (\mathcal{N}, \mathcal{E}, \mathcal{R})$, where $\mathcal{N}, \mathcal{E}$ and $\mathcal{R}$ represent the set of N nodes, the set of E typed edges, where $\mathcal{E} \subseteq \mathcal{R} \times \mathcal{N} \times \mathcal{N}$, and the set of R relations, correspondingly. Note that in this case there is no signal (i.e. feature matrix). We represent the graphs by an adjacency tensor $\boldsymbol{T} = (t_{i,j,r})_{i,j,r=1}^{N,N,R} \in \mathbb{R}^{N\times N\times R}$. **Frobenius norm.** The *weighted Frobenius norm* of a tensor where its two first dimension are equal $\boldsymbol{D} \in \mathbb{R}^{N\times N\times R}$ with respect to the *weight* $\boldsymbol{Q} \in (0,\infty)^{N\times N\times R}$ is defined to be $\|\boldsymbol{D}\|_{\mathrm{F};\boldsymbol{Q}} := \sqrt{\frac{1}{\sum_{i,j,r=1}^{N,N,R} q_{i,j,r}} \sum_{i,j,r=1}^{N,N,R} d_{i,j,r}^2 q_{i,j,r}}$.

Cut-norm. The weighted *tensor cut norm* of $\boldsymbol{D} \in \mathbb{R}^{N\times N\times R}$ with weights $\boldsymbol{Q} \in (0,\infty)^{N\times N\times R}$, is defined to be

$$\|\boldsymbol{D}\|_{\square;\boldsymbol{Q}} = \frac{1}{\sum_{i,j,r} q_{i,j,r}} \sum_{r=1}^{R} \max_{\mathcal{U},\mathcal{V}\subset[N]} \Big| \sum_{i\in\mathcal{U}} \sum_{j\in\mathcal{V}} d_{i,j,r} q_{i,j,r} \Big|.$$

and the definition for the densifying cut similarity follows.

I.2 Approximations by intersecting blocks

We define an *Intersecting Block Graph Embedding (IbgE)* with K classes (K-IbgE) as a low rank knowledge graph $\boldsymbol{C} \in \mathbb{R}^{N\times N\times R}$ with an adjacency tensor given respectively by

$$\boldsymbol{C}_{:,:,\rho} = \sum_{j=1}^{K} r_{j,\rho} \mathbb{1}_{\mathcal{U}_j} \mathbb{1}_{\mathcal{V}_j}^{\top}$$

where $\boldsymbol{r}_{j,:} \in \mathbb{R}^R$, and $\mathcal{U}_j, \mathcal{V}_j \subset [N]$. We relax the $\{0,1\}$-valued hard source/target community affiliation functions $\mathbb{1}_{\mathcal{S}}, \mathbb{1}_{\mathcal{T}}$ to soft affiliation functions in $\mathbb{R}$ to allow differentiability.

Definition I.1. *Let $d \in \mathbb{N}$, and let $\mathcal{Z}$ be a soft affiliation model. We define $[\mathcal{Z}] \subset \mathbb{R}^{N\times N\times R}$ to be the set of all elements of the form $\boldsymbol{u}\boldsymbol{v}^{\top} \otimes \boldsymbol{m}$, with $\boldsymbol{u}, \boldsymbol{v} \in \mathcal{Q}$ and $\boldsymbol{m} \in \mathbb{R}^R$. We call $[\mathcal{Z}]$ the* soft rank-1 intersecting block graph embedding (IbgE) model *corresponding to $\mathcal{Z}$. Given $K \in \mathbb{N}$, the subset $[\mathcal{Q}]_K$ of $\mathbb{R}^{N\times N\times R}$ of all linear combinations of K elements of $[\mathcal{Z}]$ is called the* soft rank-K IbgE model *corresponding to $\mathcal{Z}$.*

In matrix form, an IbgE $\boldsymbol{C} \in \mathbb{R}^{N\times N\times R}$ in $[\mathcal{Z}]_K$ can be represented by a triplet of *source community affiliation matrix* $\boldsymbol{V} \in \mathbb{R}^{N\times K}$, *target community affiliation matrix* $\boldsymbol{U} \in \mathbb{R}^{N\times K}$, and *community relations matrix* $\boldsymbol{M} \in \mathbb{R}^{K\times R}$, that satisfies:

$$\boldsymbol{C}_{:,:,\rho} = \boldsymbol{U}\,\mathrm{diag}(\boldsymbol{M}_{:,\rho})\boldsymbol{V}^{\top}$$

where $\rho \in [R]$.

I.3 Fitting IBGs to knowledge graphs

Given a knowledge graph $G = (\mathcal{N}, \mathcal{E}, \mathcal{R})$ with N nodes and R relations and an adjacency tensor $\boldsymbol{T} \in \mathbb{R}^{N\times N\times R}$ – a true triple is defined as (η, ρ, τ), where $\eta, \tau \in \mathcal{N}$ and $\rho \in [R]$. We define the

following soft rank-K intersecting block score function, based on the weighted Frobenius distance:

$$d(\eta, \rho, \tau) = \left\| \delta(\eta, \rho, \tau) - \boldsymbol{u}_{\eta,:} \operatorname{diag}(\boldsymbol{r}_{:,\rho}) \boldsymbol{v}_{:,\tau}^{\top} \right\|_{\mathrm{F};\boldsymbol{Q}_{\boldsymbol{T}}}^2$$

where the weight matrix $\boldsymbol{Q}_{\boldsymbol{T}}$ is

$$\boldsymbol{Q}_{\boldsymbol{T}} = \boldsymbol{Q}_{\boldsymbol{T},\Gamma} := e_{E,\Gamma} \mathbf{1} + (1 - e_{E,\Gamma}) \max_{\rho \in [R]} \boldsymbol{T}_{:,:,\rho},$$

$\delta(h, r, t)$ is 1 if (h, r, t) is a true triplet, otherwise 0, $\boldsymbol{V}_{\tau,:} \in \mathbb{R}^K$ is the source community affiliation of the head entity, $\boldsymbol{U}_{\eta,:} \in \mathbb{R}^K$ is the target community affiliation of the tail entity, $\boldsymbol{R}_{:,\rho} \in \mathbb{R}^K$ is the community relation vector, and $e_{E,\Gamma} = \frac{\Gamma E/N^2}{1-(E/N^2)}$ with $\Gamma > 0$.

We minimize a margin-based loss function with negative sampling, similar to [40]:

$$L = -\log \sigma \left(\gamma - d(\eta, \rho, \tau) \right) - \sum_{i=1}^{\zeta} \frac{1}{\zeta} \log \sigma \left(d\left(\eta_i', \rho_i', \tau_i'\right) - \gamma \right)$$

where γ is a fixed margin, σ is the sigmoid function, ζ is the number of negative samples and $(\eta_i', \rho_i', \tau_i')$ is the i-th negative triplet. An empirical validation of IbgE is provided in Appendix M.1.3.

J Comparison of IBGs to ICGs

In this section we highlight the key advancements and distinctions between our method and its predecessor [11].

Approximation of directed graphs. Graph directionality is crucial for accurately modeling real-world systems where relationships between entities are inherently asymmetric. Many applications rely on directed graphs to capture the flow of information, influence, or dependencies, and ignoring directionality can lead to the loss of critical structural information. One domain which benefits from directionality is spatiotemporal graphs – graphs where the topology is constant over time but the signal varies. For example, in traffic networks [4], where directionality represents the movement of vehicles along roads, traffic congestion in one direction does not necessarily imply congestion in the opposite direction. Thus, ignoring directionality can lead to inaccurate predictions. Another domain is citation networks, where nodes represent academic papers, and directed edges represent citations from one paper to another. The concrete task of prediction of the publication year is highly depedant on the direction of each edge due to the causal nature of the relation between two papers [20]. Many more additional domains benefit from directionality, some of which are Email Networks, Knowledge Graphs and Social Networks [47, 48].

ICG-NNs utilize a community affiliation matrix, that assigns to each node, community pair a value. The community affiliation is then used as a symmetric graph decomposition, which restricts the approximation to undirected graphs, resulting in their limited applicability to real-world scenarios as previously mentioned. IBG-NNs solve this issue by using source and target affiliation matrices that are used to create a general graph decomposition. This allows IBG-NNs to approximate directed graphs, making them applicable to any graph.

Densifying the adjacency matrix. The major caveat of ICG-NNs is their limited applicability to sparse graphs. Both their theoretical guarantees and approximation capabilities weaken as sparsity increases, with the approximation error of ICGs being $\mathcal{O}(N/(EK)^{1/2})$. This issue stems from the imbalance between the number of edges and non-edges in sparse graphs. As graphs grow sparser, non-edges dominate, and since standard metrics assign equal weight to edges and non-edges, the approximation shifts from capturing the relational information that we aim to capture – to capturing the absence of relations. To address this limitation, we introduce the densifying cut similarity, a novel similarity measure that explicitly accounts for the structural imbalance in sparse graphs. This similarity measure enables IBG-NNs to efficiently learn a densified representation of sparse graphs, achieving an approximation error of $\mathcal{O}(K^{-1/2})$ for both sparse and dense graphs while preserving the relational information. We emphasize that, unlike ICGs, whose approximation quality is measured with respect to the cut norm, IBGs are evaluated based on the densifying cut similarity.

Architecture and empirical performance. At first glance, ICG-NNs and IBG-NNs may appear conceptually similar, as both follow a two-step process: first estimating the graph approximation,

followed by processing through a neural network architecture. Both also support a subgraph stochastic gradient descent (SGD) method. However, the fine-grained details of their implementations differ significantly.

In the graph approximation stage, IBG-NNs extend the capabilities of ICG-NNs in two ways. They can approximate directed graphs, and more importantly they minimize a weighted Frobenius norm, where edges and non-edges are assigned different weights. This contrasts with ICG-NNs, which minimize the standard Frobenius norm. This critical difference enables IBG-NNs to handle sparse graphs more effectively, as demonstrated in Appendix M.3. In the neural network stage, IBG-NNs offer greater architectural flexibility. While ICG-NNs operate on a single community signal, IBG-NNs incorporate two community signals (source and target) when mapping back to the node space. Although we simplify the architecture by using a basic addition operation, more sophisticated manipulations could be employed. These architectural and methodological advancements result in IBG-NNs' empirical superiority across various domains. Specifically, IBG-NNs outperform ICG-NNs in node classification (see Appendix M.1.1), spatiotemporal property prediction (see Appendix M.1.2), subgraph SGD on large graphs (see Section 6.3 and Appendix M.1.4), and efficiency in the number of communities used (see Appendix M.2.1). This broad dominance underscores the effectiveness of IBG-NNs in addressing the limitations of ICG-NNs while delivering state-of-the-art performance.

Both IBG and ICG approximations capture the large scale behavior of graphs very well, since they are optimized w.r.t. the cut similarity and cut norm, which tend to ignore local small-scale structures. Hence, their approach may be less effective in domains where local small-scale structures play a key predictive role, like in estimating chemical properties of molecules. Another limitation of the methods is that the IBG and ICG are fitted to a given graph, and cannot be naturally transferred between graphs. This limits applicability to graph-level problems like graph classification.

K Complexity comparison of IBG-NN and MPNNs

Our IBG-NN architecture takes $O(D(NK + KD + ND))$ operations at each layer. For comparison, simple MPNNs such as GCN and GIN compute messages using only the features of the nodes, with computational complexity $\mathcal{O}(ED + ND^2)$. More general message-passing layers which apply an MLP to the concatenated features of the node pairs along each edge have a complexity of $\mathcal{O}(ED^2)$. Consequently, IBG neural networks are more computationally efficient than MPNNs when $K < D\mathrm{d}$, where d denotes the average node degree, and more efficient than simplified MPNNs like GCN when $K < \mathrm{d}$.

L Additional implementation details on IBG-NN

Let us recall the update equation of an IBG-NN for layers $1 \leq \ell \leq L-1$

$$\boldsymbol{H}_s^{(\ell+1)} = \sigma\left(\Theta_1^s\left(\boldsymbol{H}_s^{(\ell)}\right) + \Theta_2^s\left(\boldsymbol{V}\boldsymbol{B}^{(\ell)}\right)\right),$$

$$\boldsymbol{H}_t^{(\ell+1)} = \sigma\left(\Theta_1^t\left(\boldsymbol{H}_t^{(\ell)}\right) + \Theta_2^t\left(\boldsymbol{U}\boldsymbol{F}^{(\ell)}\right)\right),$$

And finally for layer L:

$$\boldsymbol{H}^{(L)} = \boldsymbol{H}_s^{(L)} + \boldsymbol{H}_t^{(L)},$$

In each layer, the learned functions Θ_1^s and Θ_1^t are applied to the previous node representations $\boldsymbol{H}_s^{(\ell)}$and $\boldsymbol{H}_t^{(\ell)}$ seperately, while Θ_2^s and Θ_2^t are applied to the post-analysis source community features $\boldsymbol{V}\boldsymbol{B}^{(\ell)}$ and the post-analysis target community features $\boldsymbol{U}\boldsymbol{B}^{(\ell)}$, respectively. To simplify, we restrict the aforementioned architecture of the learned function to linear layers and a pooling operation (e.g. DeepSets [49]), interleaving with a ReLU activation function. An example of an IBG-NN linear layer with mean pooling is:

$$\Theta_1\left(\boldsymbol{H}^{(\ell)}\right) = \boldsymbol{H}^{(\ell)}\boldsymbol{W}_1^{(\ell)} + \frac{1}{N}\mathbf{1}\mathbf{1}^\top \boldsymbol{H}^{(\ell)}\boldsymbol{W}_2^{(\ell)}$$

with $\boldsymbol{W}_1^{(\ell)}, \boldsymbol{W}_2^{(\ell)} \in \mathbb{R}^{D^{(\ell)} \times D^{(\ell+1)}}$ being learnable weight matrices.

IBG-NNs for spatio-temporal graphs. Given a graph with fixed connectivity and time-varying node features, we fit the IBG to the graph once. We then train a model on the frozen IBG to predict the next-step signal from past time steps. Thus, given T training signals, an IBG-NN requires $\mathcal{O}(TNKD^2)$ operations per epoch, compared to $\mathcal{O}(TED^2)$ for MPNNs, with the preprocessing time remaining independent of T. Thus, as the number of training signals increases, the efficiency gap between IBG-NNs and MPNNs becomes more pronounced.

M Additional experiments

Table 2: Results on real-world node classification datasets. Top three models are colored by First, Second, Third.

Model	Squirrel	Chameleon	Tolokers
# nodes	5201	2277	11758
# edges	217073	36101	519000
Avg. degree	41.74	15.85	88.28
Metrics	Accuracy	Accuracy	ROC AUC
MLP	28.77 ± 1.56	46.21± 2.99	72.95± 1.06
GCN	53.43 ± 2.01	64.82 ± 2.24	83.64± 0.67
GAT	40.72 ± 1.55	66.82 ± 2.56	83.70± 0.47
H_2GCN	61.9 ± 1.4	46.21 ± 2.99	73.35± 1.01
GPR-GNN	74.8± 0.5	78.3± 0.6	72.94± 0.97
LINKX	61.81 ± 1.80	68.42 ± 1.38	–
FSGNN	74.10 ± 1.89	78.27 ± 1.28	–
ACM-GCN	67.40 ± 2.21	74.76 ± 2.20	–
GloGNN	57.88 ± 1.76	71.21 ± 1.84	73.39± 1.17
Grad. Gating	64.26 ± 2.38	71.40 ± 2.38	–
DirGNN	75.13 ± 1.95	79.74 ± 1.40	–
FaberNet	76.71 ± 1.92	80.33 ± 1.19	–
ICG-NN	64.02 ± 1.67	63.9 ± 2.13	83.73± 0.78
IBG-NN	77.63± 1.79	80.15± 1.13	83.76 ± 0.75

M.1 Analysis on varying domains

M.1.1 Node classification on directed graphs

Setup. We evaluate IBG-NN on several dense directed benchmark datasets: Tolokers [50], following the 10 splits of [50], Squirrel and Chameleon [21], both following the 10 splits of [21]. We report the average ROC AUC and standard deviation for Tolokers, and the average accuracy and standard deviation for Squirrel and Chameleon. For Tolokers, we report the baselines MLP, GCN [25], GAT [51], H_2GCN [52], GPR-GNN [53], GloGNN [22] and ICG-NN [11] taken from [11]. For Squirrel and Chameleon, we report the same baselines, as well as LINKX [54], FSGNN [55], ACM-GCN [56], Grad. Grating [57], DirGNN [20] and FaberNet [27], taken from [27].

Results. Table 2 presents another domain in which IBG-NNs outperform ICG-NNs. More importantly, it establishes IBG-NNs as a state-of-the-art method for directed graphs, surpassing GNNs specifically tailored for directed graphs, despite their quadratic scaling compared to IBG-NNs.

M.1.2 Spatio-temporal graphs

Setup. We evaluate IBG-NN on the real world traffic-network datasets METR-LA and PEMS-BAY [4]. We report the baselines DCRNN [4], GraphWaveNet [58], AGCRN [59], T&S-IMP, TTS-IMP, T&S-AMP, and TTS-AMP [60], and ICG-NN [11] all taken from [11]. We follow the methodology of [60], segmenting the datasets into windows of time steps, and training the model to predict the subsequent 12 observations. Each window is divided sequentially into train, validation, and test using a $70\%/10\%/20\%$ split. We report mean average error and standard deviation over 5 different seeds. Finally, we use a GRU to embed the data before using it as input for the IBG-NN model.

[3] our codebase is publicly available at: `https://github.com/jonkouch/IBGNN-clean`

Results. Table 3 demonstrates that IBG-NNs suppresses ICG-NNs by a small margin. This slight difference could be attributed to the small size of the graph (207 and 325 nodes), where local interactions are likely sufficient for the task, and the ability of IBG-NNs to capture global structure becomes less relevant. This raises questions about the role of directionality in traffic networks, suggesting the need for further investigation. Notably, IBG-NNs show strong performance in another domain, matching the effectiveness of methods specifically designed for spatio-temporal data, such as DCRNN, GraphWaveNet, and AGCRN, despite the small graph size and low edge density.

Table 3: Results on temporal graphs. Top three models are colored by First, Second, Third.

Model	METR-LA	PEMS-BAY
# nodes	207	325
# edges	1515	2369
Avg. degree	7.32	7.29
Metrics	MAE	MAE
DCRNN	3.22 ± 0.01	1.64 ± 0.00
GraphWaveNet	3.05 ± 0.03	1.56 ± 0.01
AGCRN	3.16 ± 0.01	1.61 ± 0.00
T&S-IMP	3.35 ± 0.01	1.70 ± 0.01
TTS-IMP	3.34 ± 0.01	1.72 ± 0.00
T&S-AMP	3.22 ± 0.02	1.65 ± 0.00
TTS-AMP	3.24 ± 0.01	1.66 ± 0.00
ICG-NN	3.12 ± 0.01	1.56 ± 0.00
IBG-NN	3.10 ± 0.01	1.55 ± 0.00

M.1.3 Knowledge graphs

Setup. We evaluate IbgE on the the Kinship and UMLS [5] datasets. We report the knowledge graph embedding baselines TransE [61], DistMult [62], ConvE [13], ComplEx [63] and RotatE [40]. We also report the rule learning baselines Neural-LP [64], RNNLogic [65], RLogic [66] and NCRL [67] to compare with the state-of-the-art models over these datasets. Results for all baselines were taken from [67].

Results. Table 4 shows that IbgE achieves state of the art results across all datasets, solidifying IBGs potential as a knowledge graph embedding method. This strong performance is expected, as IbgE is both computationally efficient and expressive, capable of modeling arbitrary head-relation-tail triplets, making it particularly well-suited for modeling knowledge graphs.

M.1.4 Subgraph SGD using node sampling

Setup. In Section 6.3 we test the diode sampling SGD method proposed in Appendix H. We follow the setups described for Flickr and Reddit in Section 6.3, under condensation ratios $r = M/N$, where

Table 4: Comparison with KG completion methods. Top three models are colored by First, Second, Third.

Method	Model	Kinship			UMLS		
		MRR	Hit@1	Hit@10	MRR	Hit@1	Hit@10
KGE	TransE	0.31	0.9	84.1	0.69	52.3	89.7
	DistMult	0.35	18.9	75.5	0.39	25.6	66.9
	ComplEx	0.42	24.2	81.2	0.41	27.3	70.0
	RotatE	0.65	50.4	93.2	0.74	63.6	93.9
	IbgE	0.69	55.0	95.3	0.82	71.5	96.3
Rule Learning	Neural-LP	0.30	16.7	59.6	0.48	33.2	77.5
	DRUM	0.33	18.2	67.5	0.55	35.8	85.4
	RNNLogic	0.64	49.5	92.4	0.75	63.0	92.4
	RLogic	0.58	43.4	87.2	0.71	56.6	93.2
	NCRL	0.64	49.0	92.9	0.78	65.9	95.1

N is the total number of nodes, and M is the number of sampled nodes. For a condensation ratio of 100%, the competing methods correspond to standard GCN.

Results. Once again, Table 5 shows IBG-NN using node sampling subgraph SGD outperforms all other coarsening and condensation methods, as well as its predecessor, ICG-NN. These results are consistent with our theoretical guarantees (see Propositions H.1 and H.4), which predict a low approximation error. This demonstrates that IBG-NNs are capable of good approximations even while loading a small fraction of the graph into memory. Additionally, we observe that node-sampling and diode-sampling methods yield nearly identical empirical performance, although their approximation errors decay at different asymptotic rates (see Propositions H.1 and H.4), offering flexibility in choosing the sampling strategy based on practical considerations.

Table 5: Comparison of node-sampling subgraph SGD with coarsening methods across varying condensation ratios. Top three models are colored by First, Second, Third.

	Flickr			Reddit		
Condensation ratio	0.5%	1%	100%	0.1%	0.2%	100%
Random	44.0 ± 0.4	44.6 ± 0.2	47.2 ± 0.1	58.0 ± 2.2	66.3 ± 1.9	93.9 ± 0.0
Herding	43.9 ± 0.9	44.4 ± 0.6	47.2 ± 0.1	62.7 ± 1.0	71.0 ± 1.6	93.9 ± 0.0
DC-Graph	45.9 ± 0.1	45.8 ± 0.1	47.2 ± 0.1	89.5 ± 0.1	90.5 ± 1.2	93.9 ± 0.0
GCOND	47.1 ± 0.1	47.1 ± 0.1	47.2 ± 0.1	89.6 ± 0.7	90.1 ± 0.5	93.9 ± 0.0
SFGC	47.0 ± 0.1	47.1 ± 0.1	47.2 ± 0.1	90.0 ± 0.3	89.9 ± 0.4	93.9 ± 0.0
GC-SNTK	46.8 ± 0.1	46.5 ± 0.2	47.2 ± 0.1	–	–	–
SimGC	45.6 ± 0.4	43.8 ± 1.5	47.2 ± 0.1	91.1 ± 1.0	92.0 ± 0.3	93.9 ± 0.0
ICG-NN	50.1 ± 0.2	50.8 ± 0.1	52.7 ± 0.1	89.7 ± 1.3	90.7 ± 1.5	93.6 ± 1.2
IBG-NN	50.7 ± 0.1	51.2 ± 0.2	53.0 ± 0.1	92.3 ± 1.1	92.3 ± 0.6	94.1 ± 0.5

M.2 The effect of number of communities

M.2.1 Performance

Setup. We evaluate IBG-NN and ICG-NN on the non-sparse Squirrel and Chameleon graph [21]. We follow the 10 data splits of Pei et al. [21], Li et al. [22] for Squirrel and Chameleon reporting the accuracy and standard deviation.

Results. In Figure 4, IBG-NNs exhibit significantly improved performance compared to ICG-NNs. For a small number of communities (10) on Squirrel, IBG-NNs achieve 66% accuracy, whereas ICG-NNs achieve only 45%, further highlighting the superiority of IBG-NNs, allowing it to reach competitive performance while being efficient. The performance gap persists across both datasets, with IBG-NNs continuing to improve as the number of communities increases, whereas ICG-NNs appear to plateau. This trend can be attributed to IBG-NNs ' ability to capture directionality in the graph, allowing them to model increasingly fine-grained asymmetric structures that ICG-NNs cannot represent.

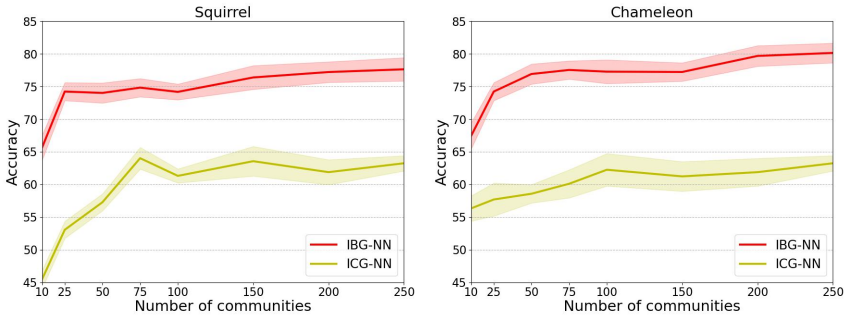


Figure 4: Accuracy of IBG-NNs and ICG-NNs on the Squirrel (**left**) and Chameleon (**right**) datasets as a function of the number of communities.

M.2.2 Approximation quality

Setup. We evaluate the effect of the number of communities K on the densifying cut similarity of the IBG approximation on the Chameleon and Squirrel datasets. We compare the weighted Frobenius error to the densifying cut similarity for a number of communities ranging from 10 to 250.

Results. As shown in Figure 5, both the densifying cut similarity and the weighted Frobenius error of IBG decrease as the number of communities increases. This stands in line with our theory, and demonstrated the bound in Equation (27) also in fact holds in practice.

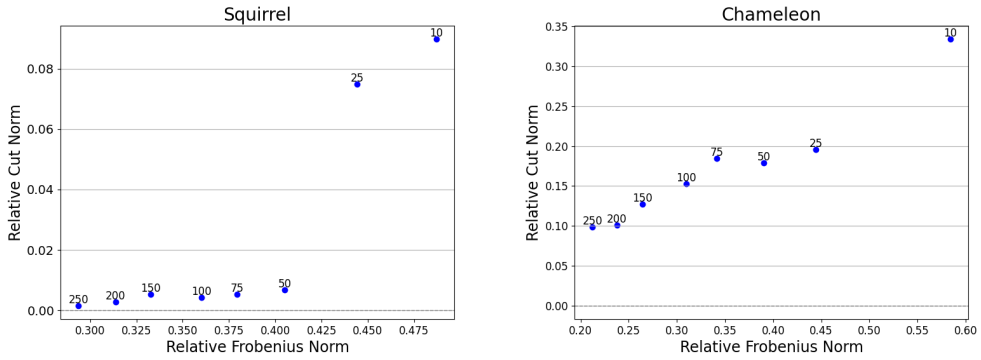


Figure 5: Densifying cut similarity as a function of the weighted Frobenius norm on the Squirrel (left) and Chameleon (right) datasets. The number of communities is specified above every point.

M.3 The importance of densifying

M.3.1 Densification for sparse graphs

Setup. We evaluate the effect of our proposed densifying lemma by assessing the performance of IBG-NN as a function of Γ, a parameter that controls the weight assigned to non-edges relative to edges in the IBG approximation. We explore Γ values ranging from 0, where weight is assigned only to existing edges, to $\frac{1-(E/N^2)}{E/N^2}$, where a uniform weight of 1 is given to each entry in the adjacency matrix. Our experiments are conducted on the large node classification graph Arxiv-Year [54] due to its low density (average degree of 6.89).

Results. Figure 6 presents that the perfromance of densified IBG-NNs with $\Gamma = \frac{1-(E/N^2)}{E/N^2}$ equals to that of IBG-NNs learned with standard unweighted loss, matching expectations. More importantly, Figure 6 shows that densified IBG-NNs consistently outperform IBG-NNs learned with standard unweighted loss, across all the densification scales, validating that the theoretical improvements also translate into significant practical benefits.

M.3.2 Effect of densification on approximation quality

Setup. We study the approximation error of ICG and IBG on their target metrics, cut metric and densifying cut similarity, when approximating *Erdős-Rényi* graphs with 1000 nodes on a different range of edge probabilities. We set the densification parameter $\Gamma = 1/2p$ when approximating $ER(1000, p)$.

Results. Figure 7 clearly demonstrates that the approximation quality of IBG remains consistent across different sparsity levels, while ICG's deteriorates as the graphs grow sparser. This greatly supports our claims that IBGs learn a densified version of the original graph while still sharing the same structure with the original graph.

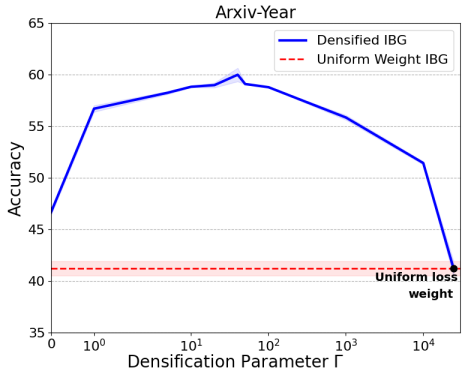


Figure 6: IBG-NN accuracy over the sparse dataset Arxiv-Year as a function of Γ. The dotted line is IBG-NN accuracy when using standard uniformly weighted loss for the IBG approximation. The rightmost point is the value of Γ which results in a uniform cut norm under the densifying loss.

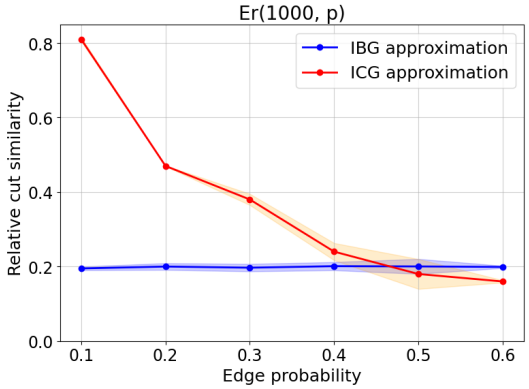


Figure 7: Cut norm and densifying cut similarity of ICG and IBG approximations on $ER(1000, p)$ for different values of p.

M.3.3 Effect of densification on IBG-NN

Setup. We compare IBG-NN and ICG-NN on the node classification dataset Squirrel when sampling only a subset of the edges with a ranging sample probability during the approximation phase.

Results. Figure 8 shows a significant performance gap increase when the graph becomes sparser. This strengthens our claims that IBG-NNs are better suited for working with sparse graphs than ICG-NNs.

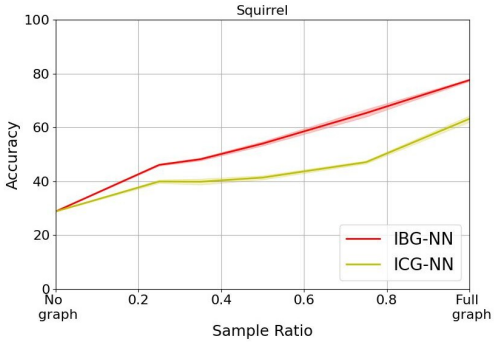


Figure 8: Node classification results on the Squirrel dataset as a function of edge sample probability during the approximation stage.

M.4 Efficiency analysis

M.4.1 Run-time analysis

Setup. We compare the forward pass runtimes of IBG-NN to that of the DirGNN [20] and FaberNet [27] methods for directed graphs on *Erdős-Rényi* $\mathrm{ER}(n, p = 0.5)$ graphs with up to $7k$ nodes. We sample 128 node features per node, independently from $U[0, 1]$. Both models use a hidden and output dimension of 128, and 3 layers.

Results. Figure 9 shows that the runtime of IBG-NN exhibits a strong square root relationship when compared to the two directed graphs methods DirGNN and FaberNet. This matches expectations, given that the time complexity of IBG-NN and message-passing neural networks are $\mathcal{O}(N)$ and $\mathcal{O}(E)$, respectively, further highlighting the difference in their efficiency.

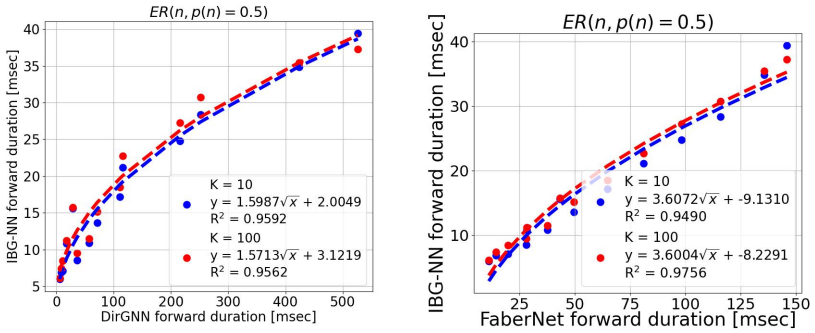


Figure 9: Runtime of K-IBG-NN as a function of DirGNN (**left**) and FaberNet (**right**) forward pass duration on $\mathrm{ER}(n, p = 0.5)$ graphs for K=10, 100.

M.4.2 Memory Analysis

Setup. We compare the IBG approximation and IBG-NN forward pass memory complexity to that of GCN. We use *Erdős-Rényi* $\mathrm{ER}(n, p = 0.5)$ graphs with up to $7k$ nodes and sample node features uniformly from $U[0, 1]$ with dimension 128. We test IBG-NN and GCN with 3 layers, and hidden and output dimensions of 128.

Results. Figure 10 clearly shows IBG's memory complexity scales linearly with GCN, while IBG-NN's memory complexity has a square root relationship with that of GCN. This result stands strongly in line with the results in Appendix M.4.1, as both the time and memory complexity of IBG approximation and IBG-NNs are $\mathcal{O}(E)$ and $\mathcal{O}(N)$ respectively.

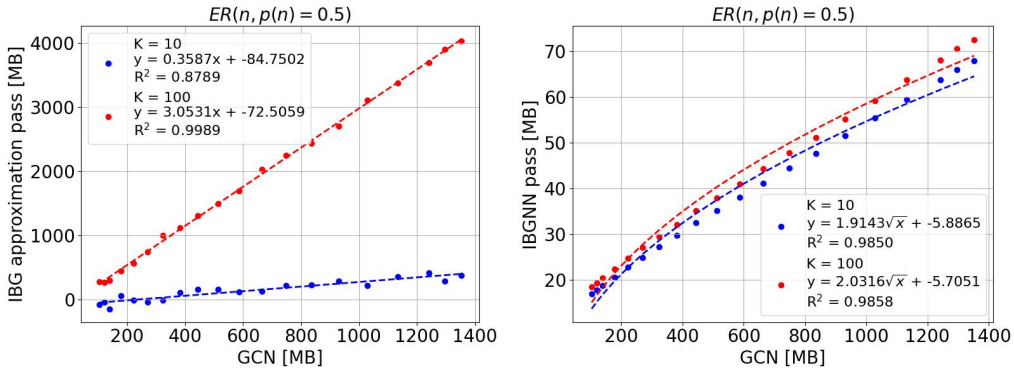


Figure 10: Memory complexity of K-IBG (left) and K-IBG-NN (right) as a function of GCN forward pass on $\mathrm{ER}(n, p = 0.5)$ graphs for K=10, 100.

N Choice of datasets

We present IBG-NNs, a method best suited for large graphs, capable of effectively approximating directional graphs. Consequently, our primary experiments in Section 6.3 follow the standard graph coarsening and condensation benchmarks Reddit [1] and Flickr [2] where IBG-NNs demonstrates state-of-the-art performance. Additionally, IBG-NNs can be viewed as a method for handling directed graphs. To evaluate its performance in this domain, we conduct experiments on standard directed graph datasets Squirrel, Chameleon [21], and Tolokers [50], where it also achieves state-of-the-art results. Finally, our graph approximation method proves particularly efficient for spatio-temporal datasets, where a fixed topology supports time-varying signals. We validate this by experimenting with IBG-NNs on the METR-LA and PEMS-BAY [4] datasets, where it matches the effectiveness of methods that are specifically designed for spatio-temporal data.

O Dataset statistics

The dataset statistics of the real-world directed graphs, spatio-temporal, graph coarsening and knowledge graph benchmarks used are presented below in Tables 6 to 9.

Table 6: Non-sparse node classification dataset statistics.

	Squirrel	Chameleon	Tolokers
# nodes (N)	5201	2277	11758
# edges (E)	217073	36101	519000
Avg. degree ($\frac{E}{N}$)	41.71	15.85	88.28
# node features	2089	2325	10
# classes	5	5	2
Metrics	Accuracy	Accuracy	ROC AUC

Table 7: Spatio-temporal dataset statistics.

	METR-LA	PEMS-BAY
# nodes (N)	207	325
# edges (E)	1515	2369
Avg. degree ($\frac{E}{N}$)	7.32	7.29
# node features	34272	52128
Metrics	MAE	MAE

Table 8: Graph coarsening dataset statistics.

	Flickr	Reddit
# nodes (N)	89250	232965
# edges (E)	899756	114615892
Avg. degree	10.08	491.99
# node features	500	602
# classes	7	41
Metrics	Accuracy	Micro-F1

Table 9: Knowledge graph completion dataset statistics.

	Kinship	UMLS
# entities	104	135
# relations	25	46
# training triples	8544	5216
# validation triples	1068	652
# testing triples	1074	661
Avg. train. degree	82.15	38.64

P Hyperparameters

In Tables 10 to 13, we report the hyper-parameters used in our real-world directed graphs, spatio-temporal, graph coarsening and knowledge graph completion benchmarks.

Table 10: Non-sparse node classification hyperparameters.

	Squirrel	Chameleon	Tolokers
# communities	50, 250	50, 250	50, 250
Encoded dim	128	128	-
β/α	1	1	1
Γ	20	5	5
Approx. lr	0.03	0.03	0.03
Approx. epochs	10000	10000	10000
# layers	5,6,7	5,6,7	4,5,6
Hidden dim	128	128	128
Dropout	0.2	0.2	0.2
Residual connection	-	✓	✓
Jumping knowledge	Cat	Cat	Max
Normalization	True	True	True
Fit lr	0.003, 0.005	0.003, 0.005	0.003, 0.005
Fit epochs	1500	1500	1500

Table 11: Spatio-temporal node regression hyperparameters.

	METR-LA	PEMS-BAY
# communities	50	100
Encoded dim	-	-
β/α	0	0
Γ	0	0
Approx. lr	0.01, 0.05	0.01, 0.05
Approx. epochs	10000	10000
# layers	5,6,7	5,6,7
Hidden dim	64, 128	64, 128
Dropout	0.0	0.0
Residual connection	✓	✓
Jumping knowledge	Max	Cat
Normalization	True	False
Fit lr	0.001, 0.003	0.001, 0.003
Fit epochs	300	300

Table 12: Graph coarsening node classification hyperparameters.

	Reddit	Flickr
# communities	50, 750	50, 750
Encoded dim	-	-
β/α	0	0
Γ	5	5
Approx. lr	0.05	0.05
Approx. epochs	0	0
# layers	2, 3, 4	2, 3, 4
Hidden dim	128, 256	128, 256
Dropout	0	0
Residual connection	✓	-
Jumping knowledge	Max	Cat
Normalization	True	True
Fit lr	0.003, 0.005	0.003, 0.005
Fit epochs	1500	1500

Table 13: Knowledge graph completion hyperparameters.

	Kinship	UMLS
# communities	20	15
Encoded dim	24	48
Approx. lr	0.05	0.003
Approx. epochs	250	750
# negative samples	64	128

For our spatio-temporal experiments, we utilize the time of the day and the one-hot encoding of the day of the week as additional features, following [60]. Additionaly, for spatio-temporal graphs we usually ignore the signal when fitting the IBG to the graph, but one can also concatenate random training signals and reduce their dimension to obtain one signal with low dimension D to be used as the target signal for the IBG optimization.