Convergence of gradient based training for linear Graph Neural Networks

Dhiraj Patel, Anton Savostianov, Michael T. Schaub
Computational Network Science, RWTH Aachen University
Aachen, Germany

Abstract

Graph Neural Networks (GNNs) are powerful tools for addressing learning problems on graph structures, with a wide range of applications in molecular biology and social networks. However, the theoretical foundations underlying their empirical performance are not well understood. In this article, we examine the convergence of gradient dynamics in the training of linear GNNs. Specifically, we prove that the gradient flow training of a linear GNN with mean squared loss converges to the global minimum at an exponential rate. The convergence rate depends explicitly on the initial weights and the graph shift operator, which we validate on synthetic datasets from well-known graph models and real-world datasets. Furthermore, we discuss the gradient flow that minimizes the total weights at the global minimum. In addition to the gradient flow, we study the convergence of linear GNNs under gradient descent training, an iterative scheme viewed as a discretization of gradient flow.

Keywords: Graph; Graph neural network; Deep learning; Node regression; Gradient flow; Optimization.

1 Introduction

Deep learning methods have been extensively applied in many applications of the machine learning field, such as computer vision [27], speech recognition [16], and natural language processing [40]. It helps identify patterns from large and complex datasets. Gradient descent is a learning paradigm of deep neural networks, used to train the networks to optimize the prediction error and significantly enhance their accuracy. While deep neural networks have shown exceptional performance in many fields, the mathematical principles underlying their success are not well understood. In this paper, we study the mathematical properties of the optimization method for deep learning performance.

The mathematical study of learning deep networks focuses on the convergence of the gradient descent algorithm, which optimizes the prediction error of the network. Non-linear deep neural networks express complex and non-linear relationships within datasets; however, the optimization problem is challenging due to their non-convex architecture. On the contrary, linear neural networks are not ideal for modeling complex data in many machine learning applications due to their simple structure. However, the non-convex loss in linear deep neural networks makes the optimization problem non-trivial. In recent years, articles

*Email id: patel@cs.rwth-aachen.de, savostianov@cs.rwth-aachen.de, schaub@cs.rwth-aachen.de
2010 *Mathematics Subject Classification.* 05C82, 91020, 92B20, 68T05.

[3, 4, 6, 14, 30, 44] have explored important properties of gradient dynamics for linear deep neural networks and studied the optimization problem in that setting.

In order to train deep networks, data are usually collected from Euclidean domains in many applications [2], such as text mining, image classification, speech recognition, and stock price prediction. However, data structures generally have complex relationships and dependencies that can be embedded in non-Euclidean domains such as graphs. Deep learning on graphs can be used to detect communities [42], predict user preferences [46], and identify the properties of molecules [45]. Graph neural networks have been gaining attention for their applications in geometric deep learning.

Graph neural networks are primarily concerned with two types of learning tasks: graph classification tasks [17, 31] and node regression or classification tasks [37, 47]. In the applications, these tasks demonstrate exceptional results. For example, the graph classification task is applied to classify various chemical compounds [43], and the node regression task is used to perform traffic forecasting [15]. However, the mathematical aspects of GNNs to comprehend their accuracy and limitations are not well studied, even less so than those of classical deep learning methods. This raises the following questions: Under what conditions does the gradient descent training of GNNs converge to the global minimum? How do we control the speed of convergence in training? How do we minimize the total weight at the global minimum? In this article, we study the convergence analysis of the gradient descent training of linear GNN, particularly for the node regression problem.

Background and related works. The convergence of the gradient descent method for a convex objective function is well understood [9, 38]. In fact, all local minima of a convex objective function are global minima. However, a sub-optimal local minimum may exist for a non-convex objective function. As a consequence, finding the global minimum of a general non-convex function by gradient descent is NP-complete [36]. In order to ensure the convergence of gradient descent to the global optimum, it is necessary to assume some additional conditions on the network. Recent theoretical studies have proven the global optimal solution for gradient-based training of the neural networks with linear activation [3, 4, 6, 47], ReLU activation [5, 35], smooth activation [12], and Gaussian inputs [11, 24].

Several researchers have concentrated their study on the critical points of objective functions to investigate the optimal training loss [23, 34]. Lee et al. [34] show that gradient descent with a random initialization converges to the local minimum if the objective function satisfies the strict saddle point property. While a shallow network always satisfies the strict saddle point property, a deep network with more than two hidden layers probably fails to satisfy such a condition [30]. On the other hand, articles [3, 6, 12, 14, 41] focus on the initialization and the training data for the convergence of the gradient descent method in deep neural networks. In the linear deep neural network, Bartlett et al. [8] prove that the gradient descent algorithm with the square loss converges to the global minimum if the initial loss is sufficiently close to the global minimum. Later, Chatterjee [12] generalizes the result for non-linear deep neural networks. The articles [3, 6] study the convergence analysis of gradient flow under the balanced conditions of the initial weight matrix.

The convergence analysis in the context of graph neural networks has not been explored much. Xu et al. [47] consider a linear graph neural network model and show that the positive singular margin of the initial weights is sufficient to ensure the square loss converges to the global minimum. Indeed, the loss surface is devoid of any sub-optimal local minima [32]. Awasthi et al. [5] generalize this result for shallow GNNs with ReLU activation and examine the convergence of gradient descent training in a probabilistic framework. In particular, with Gaussian initialization, the gradient descent algorithm recovers the realizable data of the GNN architecture with high probability.

Contributions. We present a comprehensive study on the convergence of gradient descent training for linear graph neural networks. The key features of the article are as follows.

- In a semi-supervised setting, we show that the mean square loss exponentially converges to its global minimum in a gradient flow training of linear GNNs. The convergence rate is proportional to the singular value of the initial weight matrices and the feature data embedded on graphs. Moreover, we discuss the gradient descent algorithm for linear GNN training.
- The gradient flow training of linear GNN with respect to square loss converges to the global minimum, provided the initial weight parameters have a positive singular margin [47]. In practice, verifying such a condition for any given weight matrix is challenging. We study the convergence analysis of gradient dynamics in linear GNNs without such assumptions and provide an initialization that is free from the restriction on saddle points, imposed by singular margin assumption, and ensures convergence to the global minimum.
- The convergence analysis of the gradient dynamics is discussed for deep neural networks, assuming the given data matrix is full rank [3, 6, 12], and in the context of GNN, the given feature matrix associated with the graph shift matrix is full rank [5, 47]. To the best of our knowledge, the convergence analysis of gradient-based training in deep neural networks associated with low-rank data has not been studied in the literature. In this article, we show that the convergence of the mean square loss to the global minimum depends on the smallest non-zero singular value of the product of the feature data and graph shift matrix.
- Under the "nice" initialization, the gradient-based training converges to the global minimum of the loss surface. However, based on the GNN architecture, the loss surface might have infinitely many global minima. Minimizing the weights at the global minima reduces the computational complexity of the model and compresses the memory storage while the model remains consistent in terms of its accuracy and performance. In this context, we explore the properties of initialization, which leads the training process towards minimizing the loss and optimizing the weights.

Outline. The paper is organized as follows. Linear GNNs are defined in Section 2, which discusses the preliminary results of matrix theory and the mean square loss associated with linear GNN. Section 3 explores the convergence analysis of gradient flow training for linear GNN, while the optimization of the weights at the global minima of the loss surface is discussed in Section 4. The analogous results for the convergence of gradient flow in discrete time are addressed in Section 5. Finally, in Section 6, we validate the gradient dynamics on synthetic graph datasets and a real-world graph dataset, and we discuss the limitations of our result.

2 Preliminaries and problem setup

In this section, we recall the notion of graphs and set up the graph neural network model. We then recap some preliminary results from matrix theory that are used later in the paper.

2.1 Graphs and Graph Neural Networks

Graphs and data supported on graphs. A graph is defined as an ordered pair $G = (V, E)$, which consists of a set of nodes V and a set of edges $E \subseteq \{\{u, v\} \mid u, v \in V\}$

describing pairwise relations between the nodes. For simplicity, we identify the node set with the integers from 1 up to n as follows, i.e., $V = \{1, \dots, n\}$, where n is the number of nodes in the graph. This enables us to encode the structure of the graph into an adjacency matrix $A \in \{0,1\}^{n\times n}$ with entries $A_{ij} = 1$ if the edge $\{i, j\}$ exists in the edge set of the graph, and $A_{ij} = 0$, otherwise. We say a matrix $S \in \mathbb{R}^{n\times n}$ is a graph shift operator if it has the same sparsity pattern as A in the off-diagonal terms, i.e., we have $S_{ij} \neq 0 \Leftrightarrow A_{ij} \neq 0$ for all $i \neq j$.

In the context of machine learning on graphs, we are interested in semi-supervised learning, where each node in the graph has feature data associated with it, and the graph contains both labeled and unlabeled nodes. We are concerned with predicting the labels of the nodes based on the graph structure and node features. We assume that each node i is endowed with a feature vector $x_i \in \mathbb{R}^{d_x}$ (describing, e.g., certain attributes of the node), and $X \in \mathbb{R}^{d_x\times n}$ denotes the feature data matrix with the i-th columns corresponding to feature data x_i. The set $\mathcal{I} \subset V$ is the collection of labeled nodes and $y_j \in \mathbb{R}^{d_y}$ (describing quantities we may want to estimate or predict) is the label vector for node $j \in \mathcal{I}$. The labeled node matrix is denoted by $Y = [y_i]_{i\in\mathcal{I}} \in \mathbb{R}^{d_y\times\bar{n}}$, where $\bar{n}$ is the number of labeled nodes.

Graph Neural Networks. Graph neural network is a mapping from node feature data to label data, and it operates by transmitting and aggregating messages between graph nodes. Our problem of interest is to train GNN to improve its accuracy in the prediction of label data.

The architecture of deep GNN is composed of several hidden layers, each containing node data derived from the aggregation of the node data from the previous layer. Mathematically, if X_ℓ denotes the collection of node data on the ℓ-th layer, then data representation on $(\ell+1)$-th layer is given by

$$\begin{aligned} Z_\ell &= F(X_\ell) \\ X_{\ell+1} &= \varphi_\ell(Z_\ell), \end{aligned}$$

where F denotes the linear filter equipped with trainable weights, and $\varphi_\ell : \mathbb{R} \to \mathbb{R}$ is an activation function that acts componentwise. In particular, graph convolution filter is defined by

$$F(X) = \sum_{k=1}^{K} W_k X S^k,$$

where S is the graph aggregate matrix, and W_k's are trainable weights. GNNs with convolution filters $K \leq 1$ are referred to as message passing networks, since the node representation depends exclusively on its immediate neighbors. In semi-supervised learning, Kipf and Willing [37] discussed the effectiveness of a message passing network with the ReLU activation function. In this article, we explore the mathematical properties of message passing network optimization using linear activation functions.

Definition 2.1 (Linear GNN)**.** Let $G = (V, E)$ be a graph and $S \in \mathbb{R}^{n\times n}$ denote the corresponding graph shift matrix. The linear GNN with H layers is a map $f : \mathbb{R}^{d_x\times n} \to \mathbb{R}^{d_y\times n}$ such that

$$f(X) = W_{H+1}X_H, \qquad \text{and} \qquad X_\ell = W_\ell X_{\ell-1}S, \quad \text{for } 1 \leq \ell \leq H, \tag{2.1}$$

where $W_1, \dots, W_{H+1}$ represent the weight matrices of the corresponding order during propagation, and X_0 denotes the feature matrix X.

The output of the linear GNN depends on the collection of weight matrices $\boldsymbol{W} = (W_1, W_2, \dots, W_{H+1})$, and our aim is to improve the accuracy of the output by appropriately choosing the weight matrices. For simplicity, we denote the map for linear GNN by

$f(\cdot, \boldsymbol{W})$, where the trainable weights $W_\ell \in \mathbb{R}^{d_\ell \times d_{\ell-1}}$ for $1 \le \ell \le H+1$, $d_0 = d_x$, and $d_{H+1} = d_y$.

The linear GNN of layer H can be viewed as an iterative model that gathers node information from $\hat{X}_{\ell-1} = X_{\ell-1}S$ from its neighbors and encodes it into a node representation $X_\ell = W_\ell \hat{X}_{\ell-1}$. More specifically, the node representation of a given node extracts contextual information from its H-hop neighborhood. The architecture of GNNs relies on the way data is collected from neighboring nodes, namely through different forms of an aggregation matrix. The adjacency matrix of the graph is one of the most popular choices for an aggregation operator. However, the adjacency matrix aggregates the node features of all neighboring nodes, which can lead to optimization difficulty and numerical instability for graphs with large variations in the degrees of nodes. One approach to address this issue is to consider $S = \hat{D}^{-1}(I_n + A)$ [25], where $\hat{D}$ is the degree matrix of the self-loop adjacency matrix $I_n + A$ of the graph G. Such aggregator matrix reduces computation costs by sampling a fixed number of neighbors for aggregation at random. The aggregation operator $S = (I_n + A)\hat{D}^{-1}$ is considered in the PageRank algorithm [10, 22], which revises the ranking of a page by computing the weighted sum of the rankings of its linked pages. In a citation graph, information from the most cited papers may be insufficient for categorizing the paper, as multiple articles reference these papers across a wide range of subfields. Kipf and Welling [37] consider the aggregation matrix $S = \hat{D}^{-\frac{1}{2}}(I_n + A)\hat{D}^{-\frac{1}{2}}$ for smoothing the "graph signal". We refer the reader to [18, 26] for a more detailed discussion on the choice of the aggregation matrix.

In order to improve the prediction of the output label data, the linear GNN is trained to minimize a loss function. Various loss functions are considered in GNN training based on different network applications. For instance, cross-entropy loss is considered for the node classification problem [33, 49], Quasi-Wasserstein loss is used for the optimal transportation problem [13, 20, 21], and mean squared loss is applied for the regression task [5, 39].

In the semi-supervised node regression problem, a GNN is not only trained to appropriately predict the accessible label data on some subset of the node set but also to efficiently predict for unlabeled nodes. In this article, we study the semi-supervised node regression problem for a linear GNN. In particular, we discuss the following minimization problem. For a given input data matrix $X \in \mathbb{R}^{d_x \times n}$ and output label matrix $Y \in \mathbb{R}^{d_y \times \bar{n}}$,

$$\tilde{\mathcal{L}}_H = \inf_{\boldsymbol{W}} \mathcal{L}(\boldsymbol{W}) := \inf_{\boldsymbol{W}} \frac{1}{m} \|f(X, \boldsymbol{W})_{*\mathcal{I}} - Y\|_F^2. \tag{2.2}$$

We assume that the label data is known on some nodes $\mathcal{I} \subset V$ with cardinality $\bar{n}$. Moreover, $f(X, \boldsymbol{W})_{*\mathcal{I}}$ denotes the predicted label data matrix on labeled nodes, and $m = \bar{n} d_y$.

For any $H \in \mathbb{N}$, the optimization problem defined in (2.2) is non-convex. However, the network $f(X, \boldsymbol{W})_{*\mathcal{I}} = W_{[1:H+1]}(XS^H)_{*\mathcal{I}}$ with the factorization

$$W_{[1:H+1]} = W_{H+1} \cdot W_H \cdots W_1$$

can be viewed as an overparameterization of the matrix $W_{[1:H+1]}$. The factorization imposes an addition constraint that the rank of $W_{[1:H+1]}$ is at most $k = \min\{d_x, d_1, d_2, \ldots, d_H, d_y\}$. This implies,

$$\inf_{\boldsymbol{W}} \mathcal{L}(\boldsymbol{W}) \ge \inf_{W \in \mathbb{R}^{d_y \times d_x}} \frac{1}{m} \|W(XS^H)_{*\mathcal{I}} - Y\|_F^2.$$

Assumption. We consider the linear GNN model with the hidden feature dimensions are in non-increasing order, i.e., $d_1 \ge d_2 \ge \cdots \ge d_{H+1}$.

For a given linear operator $R : \mathbb{R}^{d'} \to \mathbb{R}^d$, the map $\mathcal{P}_R : \mathbb{R}^d \to \mathbb{R}^d$ denotes an orthogonal projection onto the column space of R and defined by $\mathcal{P}_R := R \cdot R^\dagger$, where $R^\dagger$ denotes the Moore–Penrose inverse of R. The projection map is used to estimate the least square solution, which subsequently helps to calculate the lower bound for (2.2).

Example 2.2. For $W \in \mathbb{R}^{d_y \times d_x}$, the loss $\mathcal{L}_1(W) = \|W(XS^H)_{*\mathcal{I}} - Y\|_F^2$ is a convex function, and the least square solution to the square loss $\mathcal{L}_1$ can be explicitly determined. In particular, we have

$$\begin{aligned}\inf_{W \in \mathbb{R}^{d_y \times d_x}} \|W(XS^H)_{*\mathcal{I}} - Y\|_F^2 &= \inf_{W \in \mathbb{R}^{d_y \times d_x}} \big\| \big[(XS^H)_{*\mathcal{I}}^\top \otimes I_{d_y}\big] \mathrm{vec}\big(W\big) - \mathrm{vec}\big(Y\big) \big\|_2^2 \\ &= \big\| \mathcal{P}_{[(XS^H)_{*\mathcal{I}}^\top \otimes I_{d_y}]} \mathrm{vec}\big(Y\big) - \mathrm{vec}\big(Y\big) \big\|_2^2.\end{aligned}$$

The first equality follows from a well-known result in matrix theory [48], which states that for any $P \in \mathbb{R}^{d_P \times d'_P}$, $Q \in \mathbb{R}^{d_Q \times d'_Q}$ and $X \in \mathbb{R}^{d'_P \times d_Q}$,

$$\mathrm{vec}\big(PXQ\big) = \big(Q^\top \otimes P\big)\mathrm{vec}\big(X\big), \tag{2.3}$$

where $\otimes$ denotes the Kronecker product of matrices.

2.2 Preliminaries on matrix theory

Graph Neural Networks (GNNs) are trained to minimize the prediction loss $\mathcal{L}(\boldsymbol{W})$. The primary goal of this paper is to investigate the convergence of gradient-based training for linear GNNs to estimate $\tilde{\boldsymbol{W}}$, the parameter that optimizes the loss $\mathcal{L}$. To delve deeper into the training of linear GNNs, we first recall some preliminary results from matrix theory.

The following lemmas outline basic properties of full rank matrices, which are used in the main result on gradient flow convergence. While these results are straightforward to derive, we provide complete proofs to ensure our exposition is self-contained.

Lemma 2.3. *Let $P \in \mathbb{R}^{d \times d'}$ be a full rank matrix. Then, for any matrix $\tilde{P} \in \mathbb{R}^{d \times d'}$ with Frobenius norm $\|\tilde{P}\|_F < \sigma_{\min}(P)$, the matrix $(P + \tilde{P})$ is also full rank. Moreover, the smallest singular value of the sum of the matrices $(P + \tilde{P})$ is bounded below by $\big(\sigma_{\min}(P) - \|\tilde{P}\|_F\big)$.*

Here, $\sigma_{\min}(M)$ denotes the smallest singular value of the matrix M.

The lemma precisely states that every matrix in an open neighborhood of a full rank matrix is also a full rank. In particular, the set of all full rank matrices of the same order is an open set with respect to the Frobenius norm.

Proof. The proof of the lemma easily follows from the Weyl's inequality for the singular value of matrices [29]. For any $P, \tilde{P} \in \mathbb{R}^{d \times d'}$, we have

$$\sigma_i(P + \tilde{P}) \geq \sigma_i(P) - \|\tilde{P}\|, \qquad \text{and} \qquad \|\tilde{P}\| \leq \|\tilde{P}\|_F, \tag{2.4}$$

where $\sigma_i(P)$ and $\|\tilde{P}\|$ denote the i-th singular value of P and the spectral norm of $\tilde{P}$ respectively. □

A full rank matrix preserves the structural information of the input data and avoids the loss of dimensionality redundancy. In a linear deep neural network, data are processed with linear transformation at each layer. A low rank transformation at certain layer might lose critical information about the data. The following lemma emphasizes the critical role of full rank matrices in maintaining structural integrity and minimizing information loss during successive linear transformations.

Lemma 2.4. *Suppose $P \in \mathbb{R}^{d \times d'}$ and $Q \in \mathbb{R}^{d' \times d''}$ are full rank matrices, with $d \geq d' \geq d''$. Then, the product PQ is a full rank matrix and*

$$\sigma_{\min}(PQ) \geq \sigma_{\min}(P)\sigma_{\min}(Q).$$

Proof. The fact that P and Q are full rank matrices implies that $\text{rank}(P) = d'$, and $\text{rank}(Q) = d''$. Using Sylvester's rank inequality, we get PQ is full rank as

$$d'' \geq \text{rank}(PQ) \geq \text{rank}(P) + \text{rank}(Q) - d' = d''.$$

Let $x \in \mathbb{R}^{d''}$ be arbitrary non-zero column matrix. Then x is in row space of Q, and Qx is in row space of P. Hence,

$$\|PQx\|_2 \geq \sigma_{\min}(P)\|Qx\|_2 \geq \sigma_{\min}(P)\sigma_{\min}(Q)\|x\|_2.$$

This implies, $\sigma_{\min}(PQ) \geq \sigma_{\min}(P)\sigma_{\min}(Q.)$. □

Lemma 2.5. *Let $R \in \mathbb{R}^{d\times d'}$ be a real matrix. For every $x \in \mathbb{R}^{d'}$, we have*

$$\|Rx\|_2 \geq \sigma_{\text{small}}(R)\, \|\mathcal{P}_{R^\top} x\|_2,$$

where $\sigma_{\text{small}}(R)$ denotes the smallest non-zero singular value of R.

Proof. Let $R = U\Sigma V^\top$ be the singular value decomposition of R, where $U \in \mathbb{R}^{d\times d}$ and $V \in \mathbb{R}^{d'\times d'}$ are the orthonormal matrices, and $\Sigma \in \mathbb{R}^{d\times d'}$ is the rectangular diagonal matrix with singular values of R as the diagonal entries.

Let $x \in \mathbb{R}^{d'}$ be arbitrary. Then we have

$$\begin{aligned}
\|Rx\|_2^2 &= \|\Sigma V^\top x\|_2^2 \\
&= x^\top V \Sigma^\top \Sigma V^\top x \\
&\geq \sigma_k^2(R)\, x^\top V \Sigma^\top (\Sigma^\dagger)^\top V^\top x \\
&= \sigma_k^2(R)\, \langle \mathcal{P}_{R^\top} x, x\rangle = \sigma_k^2(R)\, \|\mathcal{P}_{R^\top} x\|_2^2.
\end{aligned}$$

This completes the proof. □

3 Convergence analysis of gradient flow training

In this section, we discuss the gradient flow training of linear GNNs and show that the gradient flow (gradient descent with infinitesimal steps) of means square loss $\mathcal{L}$ converges to the global minimum. The gradient flow training of linear GNNs evolves the weight matrices at time t as follows

$$\frac{dW_\ell(t)}{dt} = -\frac{\partial \mathcal{L}\big(\boldsymbol{W}(t)\big)}{\partial W_\ell}, \qquad 1 \leq \ell \leq H+1, \tag{3.1}$$

where $W_\ell(t)$ represents the trainable parameters at time t with initialization $W_\ell(0)$ and $\boldsymbol{W}(t) = \big(W_1(t), W_2(t), \ldots, W_{H+1}(t)\big)$.

The optimization of gradient flow training for the classical neural network has been studied in [3, 6, 12, 14] and has been generalized to GNNs [5, 47]. The main result of the paper regarding gradient flow training of linear GNNs is stated as follows.

Theorem 3.1. *Let $G = (V, E)$ be a graph embedded with a feature matrix $X \in \mathbb{R}^{d_x\times n}$, and a labeled matrix $Y \in \mathbb{R}^{d_y\times \bar{n}}$. Let us consider a linear GNN with H layers and non-increasing hidden feature dimensions. The gradient flow training (3.1) of the linear GNN with the loss function $\mathcal{L}$ defined in (2.2) converges to the global minimum under an appropriately chosen initialization.*

In particular, suppose the initial weight $W_1(0)$ is the zero matrix and $W_\ell(0)$ is full rank for $2 \leq \ell \leq H+1$. Then, with the initialization $\boldsymbol{W}(0)$:

$$\mathcal{L}\big(\boldsymbol{W}(T)\big) - \tilde{\mathcal{L}}_H \leq e^{-\frac{1}{m}\beta\sigma^2_{\text{small}}((XS^H)_{*\mathcal{I}})T}\big(\mathcal{L}(\boldsymbol{W}(0)) - \tilde{\mathcal{L}}_H\big),$$

provided $\sigma_{\min}\big(W_\ell(0)\big)$ *is sufficiently large for some* $2 \le \ell \le H+1$. *Here* $\sigma_{\text{small}}((XS^H)_{*\mathcal{I}})$ *denotes the smallest non-zero singular value of the matrix* $(XS^H)_{*\mathcal{I}}$, $\beta = \frac{1}{4^{H-1}} \prod_{\ell=2}^{H+1} \sigma^2_{\min}(W_\ell(0))$ *and* $m = \bar{n} d_y$.

The proof of Theorem 3.1 is based on the following local optimization result.

Theorem 3.2 ([12]). *Let* p *be a positive integer, and* $F : \mathbb{R}^p \to [0, \infty)$ *be a non-negative* C^2 *function. For any* $x_0 \in \mathbb{R}^p$, *if there exists* $r > 0$ *such that*

$$4F(x_0) < r^2 \inf_{x \in B(x_0, r), F(x) \neq 0} \frac{|\nabla F(x)|^2}{F(x)}, \tag{3.2}$$

then the gradient flow equation $\frac{d}{dt}\phi(t) = -\nabla F\big(\phi(t)\big)$ *has a unique solution* $\phi : [0, \infty) \to \mathbb{R}^p$ *with the initialization* $\phi(0) = x_0$. *Here,* $B(x_0, r)$ *is the closed Euclidean ball of radius* r *centered at* x_0.

Moreover, the solution ϕ *stays in* $B(x_0, r)$ *for all* $t > 0$, *and converges to some* $\tilde{x} \in B(x_0, r)$ *where* $F(\tilde{x}) = 0$. *In particular, for* $\alpha := \inf_{x \in B(x_0, r), F(x) \neq 0} |\nabla F(x)|^2 / F(x)$, *and* $t \ge 0$, *we have*

$$\|\phi(t) - \tilde{x}\|_2 \le r e^{-\alpha t/2}, \qquad \text{and} \qquad F\big(\phi(t)\big) \le e^{-\alpha t} F(x_0).$$

The above theorem characterize the convergence of gradient flow for the twice differentiable non-negative functions. The inequality (3.2) ensure the absence of saddle points in the neighborhood of the initialization and the gradient flow always stays in that neighborhood. Moreover, the initial points is appropriately chosen close to the global minimum. In the following, utilizing the idea of the Theorem 3.2, we prove the convergence result for the linear GNNs.

Proof of Theorem 3.1. The idea of the convergence result is as follows.

- First consider the initial weights such that there does not exist any critical points other than the points of global minimum in the neighborhood of initial weights. In other words, if $\boldsymbol{W}(0)$ denotes the collection of initial weights then

$$\inf \left\{ \frac{|\nabla L(\boldsymbol{W})|^2}{L(\boldsymbol{W})} : \sum_{\ell=1}^{H+1} \|W_\ell - W_\ell(0)\|_F^2 \le r^2 \right\} > 0,$$

  where $|\nabla L(\boldsymbol{W})|^2 = \sum_{\ell=1}^{H+1} \|\nabla_{W_\ell} L(\boldsymbol{W})\|_F^2$, and the loss function L is defined by

$$L(\boldsymbol{W}) = \mathcal{L}(\boldsymbol{W}) - \tilde{\mathcal{L}}_H. \tag{3.3}$$

- Second steps of the prove is to ensure that the gradient flow always stay in that neighborhood of the initial weights. In particular, the analogous condition of (3.2) for the linear GNNs, i.e.,

$$4L\big(\boldsymbol{W}(0)\big) < r^2 \inf \left\{ \frac{|\nabla L(\boldsymbol{W})|^2}{L(\boldsymbol{W})} : \sum_{\ell=1}^{H+1} \|W_\ell - W_\ell(0)\|_F^2 \le r^2 \right\}.$$

Let $\boldsymbol{W}(0)$ be the collection of the initial weights such that $W_1(0) \in \mathbb{R}^{d_1 \times d_x}$ is a zero matrix, and $W_\ell(0) \in \mathbb{R}^{d_\ell \times d_{\ell-1}}$ is full rank for $2 \le \ell \le H+1$.

The lower bound of $\frac{\|\nabla_{\mathrm{vec}(W_1)} L(\boldsymbol{W})\|_2^2}{L(\boldsymbol{W})}$ in the neighborhood of the initial weights is a lower bound of $\frac{|\nabla L(\boldsymbol{W})|^2}{L(\boldsymbol{W})}$ as well. Hence, for any collection of weights $\boldsymbol{W}$, $\nabla_{\mathrm{vec}(W_1)} L(\boldsymbol{W})$ is estimated by

$$\nabla_{\mathrm{vec}(W_1)} L(\boldsymbol{W}) = \left(\frac{\partial L}{\partial \mathrm{vec}(\hat{Y})} \frac{\partial \mathrm{vec}(\hat{Y})}{\partial \mathrm{vec}\big((X_H)_{*\mathcal{I}}\big)} \frac{\partial \mathrm{vec}\big((X_H)_{*\mathcal{I}}\big)}{\partial \mathrm{vec}(X_{H-1})} \frac{\partial \mathrm{vec}(X_{H-1})}{\partial \mathrm{vec}(X_{H-2})} \cdots \frac{\partial \mathrm{vec}(X_1)}{\partial \mathrm{vec}(W_1)} \right)^\top$$

where

$$\begin{aligned} \hat{Y} = f(X, \boldsymbol{W})_{*\mathcal{I}} &= W_{H+1}(X_H)_{*\mathcal{I}} \in \mathbb{R}^{d_y \times \bar{n}}, \\ (X_H)_{*\mathcal{I}} &= W_\ell X_{H-1}(S)_{*\mathcal{I}} \in \mathbb{R}^{d_H \times \bar{n}}, \\ X_\ell &= W_\ell X_{\ell-1} S \in \mathbb{R}^{d_\ell \times n}, \qquad \text{for } 1 \le \ell \le H-1. \end{aligned}$$

The partial derivative can be calculated by the relation (2.3), and we get

$$\begin{aligned} \nabla_{\mathrm{vec}(W_1)} L(\boldsymbol{W}) &= \Big(\frac{\partial L}{\partial \mathrm{vec}(\hat{Y})} \big[I_{\bar{n}} \otimes W_{H+1}\big] \big[(S)_{*\mathcal{I}}^\top \otimes W_H\big] \big[S^\top \otimes W_{H-1}\big] \cdots \\ &\qquad \big[S^\top \otimes W_2\big] \big[(XS)^\top \otimes I_{d_1}\big] \Big)^\top \\ &= \frac{2}{m} \big[(XS^H)_{*\mathcal{I}} \otimes W_2^\top W_3^\top \cdots W_{H+1}^\top\big] \mathrm{vec}(\hat{Y} - Y) \\ \nabla_{\mathrm{vec}(W_1)} L(\boldsymbol{W}) &= \frac{2}{m} \big[I_{d_x} \otimes W_2^\top W_3^\top \cdots W_{H+1}^\top\big] \mathrm{vec}\big((\hat{Y} - Y)(XS^H)_{*\mathcal{I}}^\top\big). \end{aligned} \tag{3.4}$$

The product $W_2^\top W_3^\top \cdots W_{H+1}^\top \in \mathbb{R}^{d_1 \times d_y}$ and $d_1 \ge d_y$ implies

$$\begin{aligned} \|\nabla_{\mathrm{vec}(W_1)} L(\boldsymbol{W})\|_2^2 &\ge \frac{4}{m^2} \sigma_{\min}^2\big(W_2^\top W_3^\top \cdots W_{H+1}^\top\big) \big\|\mathrm{vec}\big((\hat{Y} - Y)(XS^H)_{*\mathcal{I}}^\top\big)\big\|_2^2 \\ &\ge \frac{4}{m^2} \sigma_{\min}^2\big(W_2^\top W_3^\top \cdots W_{H+1}^\top\big) \big\|\big[(XS^H)_{*\mathcal{I}} \otimes I_{d_y}\big] \mathrm{vec}(\hat{Y} - Y)\big\|_2^2. \end{aligned} \tag{3.5}$$

Since $W_\ell(0) \in \mathbb{R}^{d_\ell \times d_{\ell-1}}$ is a full rank with $d_\ell \le d_{\ell-1}$ for $2 \le \ell \le H+1$, then Lemma 2.3 implies for each $W_\ell \in \mathbb{R}^{d_\ell \times d_{\ell-1}}$ with

$$\|W_\ell - W_\ell(0)\|_F \le r_\ell := \frac{\sigma_{\min}\big(W_\ell(0)\big)}{2}, \qquad 2 \le \ell \le H+1,$$

W_ℓ is full rank with $\sigma_{\min}(W_\ell) \ge \sigma_{\min}\big(W_\ell(0)\big)/2$. Moreover, Lemma 2.4 implies $\big(W_2^\top W_3^\top \cdots W_{H+1}^\top\big)$ is full rank with

$$\sigma_{\min}^2\big(W_2^\top W_3^\top \cdots W_{H+1}^\top\big) \ge \prod_{\ell=2}^{H+1} \sigma_{\min}^2(W_\ell) \ge \frac{1}{4^H} \prod_{\ell=2}^{H+1} \sigma_{\min}^2\big(W_\ell(0)\big). \tag{3.6}$$

Now, the Lemma 2.5 implies

$$\begin{aligned} \big\|\big[(XS^H)_{*\mathcal{I}} \otimes I_{d_y}\big] \mathrm{vec}(\hat{Y} - Y)\big\|_2^2 &\ge \sigma_{\mathrm{small}}^2\big(\big[(XS^H)_{*\mathcal{I}} \otimes I_{d_y}\big]\big) \|\mathcal{P}_{[(XS^H)_{*\mathcal{I}}^\top \otimes I_{d_y}]} \mathrm{vec}(\hat{Y} - Y)\|_2^2 \\ &= \sigma_{\mathrm{small}}^2\big((XS^H)_{*\mathcal{I}}\big) \big\|\mathrm{vec}(\hat{Y}) - \mathcal{P}_{[(XS^H)_{*\mathcal{I}}^\top \otimes I_{d_y}]} \mathrm{vec}(Y)\big\|_2^2 \\ &\ge \sigma_{\mathrm{small}}^2\big((XS^H)_{*\mathcal{I}}\big) \Big(\|\mathrm{vec}(\hat{Y} - Y)\|_2^2 \\ &\qquad - \big\|\mathrm{vec}(Y) - \mathcal{P}_{[(XS^H)_{*\mathcal{I}}^\top \otimes I_{d_y}]} \mathrm{vec}(Y)\big\|_2^2 \Big) \\ &\ge m \sigma_{\mathrm{small}}^2\big((XS^H)_{*\mathcal{I}}\big) \big(\mathcal{L}(\boldsymbol{W}) - \tilde{\mathcal{L}}_H\big). \end{aligned}$$

Here $\mathcal{P}_{[(XS^H)_{*\mathcal{I}}^\top \otimes I_{d_y}]}$ denote the orthogonal projection onto column space of $\big[(XS^H)_{*\mathcal{I}}^\top \otimes I_{d_y}\big]$.

Let $\mathrm{vec}\big(\boldsymbol{W}\big) \in \mathbb{R}^p$ denotes the collection $\boldsymbol{W}$ in vector form and $B\big(\mathrm{vec}\big(\boldsymbol{W}(0)\big), r\big)$ be the closed ball of radius r centered at $\mathrm{vec}\big(\boldsymbol{W}(0)\big)$ in $\mathbb{R}^p$, where $p = \sum_{\ell=1}^{H+1} d_\ell d_{\ell-1}$ and $r = \min\{r_\ell : 2 \le \ell \le H+1\}$. For each $\mathrm{vec}\big(\boldsymbol{W}\big) \in B\big(\mathrm{vec}\big(\boldsymbol{W}(0)\big), r\big)$,

$$\|W_\ell - W_\ell(0)\|_F \le r, \qquad \text{for } 1 \le \ell \le H+1.$$

Hence, for any $\mathrm{vec}\big(\boldsymbol{W}\big) \in B\big(\mathrm{vec}\big(\boldsymbol{W}(0)\big), r\big)$, the inequalities (3.5) and (3.6) implies

$$\frac{\|\nabla_{\mathrm{vec}\left(W_1\right)} L(\boldsymbol{W})\|_2^2}{L(\boldsymbol{W})} \ge \frac{1}{m}\beta\sigma_{\mathrm{small}}^2\big((XS^H)_{*\mathcal{I}}\big) \tag{3.7}$$

where $\beta = \frac{1}{4^{H-1}} \prod_{\ell=2}^{H+1} \sigma_{\min}^2(W_\ell(0))$.

Without loss of generality, assume that $\sigma_{\min}(W_{H+1}(0))$ is sufficiently large such that

$$\frac{4}{r^2} L\big(\boldsymbol{W}(0)\big) \le \frac{1}{m}\beta\sigma_{\mathrm{small}}^2\big((XS^H)_{*\mathcal{I}}\big).$$

This implies

$$4L\big(\boldsymbol{W}(0)\big) \le r^2 \inf\Big\{\frac{|\nabla L(\boldsymbol{W})|^2}{L(\boldsymbol{W})} : \mathrm{vec}\big(\boldsymbol{W}\big) \in B\big(\mathrm{vec}\big(\boldsymbol{W}(0)\big), r\big)\Big\}. \tag{3.8}$$

Consequently, Theorem 3.2 signifies the gradient flow started at $\boldsymbol{W}(0)$ converges to some $\tilde{\boldsymbol{W}}$ such that $L(\tilde{\boldsymbol{W}}) = 0$. Moreover, at time $T > 0$,

$$\sum_{\ell=1}^{H+1} \|W_\ell(T) - \tilde{W}_\ell\|_F^2 \le r^2 \exp\Big(-\frac{1}{m}\beta\sigma_{\mathrm{small}}^2((XS^H)_{*\mathcal{I}})T\Big),$$

and

$$\mathcal{L}\big(\boldsymbol{W}(T)\big) - \tilde{\mathcal{L}}_H \le e^{-\frac{1}{m}\beta\sigma_{\mathrm{small}}^2((XS^H)_{*\mathcal{I}})T}\big(\mathcal{L}(\boldsymbol{W}(0)) - \tilde{\mathcal{L}}_H\big).$$

This completes the proof. □

Remark 3.3. The convergence rate of the gradient flow training of linear GNNs explicitly depends on the singular value of weight parameters. Let $\boldsymbol{W}(0)$ be the collection of initial weight matrices such that

$$\begin{aligned} W_1(0) &: \text{ zero matrix,} \\ W_\ell(0) &: \text{ diagonal matrix with entries are 1, for } 2 \le \ell \le H, \\ W_{H+1}(0) &: \text{ diagonal matrix with entries are } a, \end{aligned}$$

then $\beta = \frac{a^2}{4^{H-1}}$. Moreover, with the initialization, the gradient flow converges to the global minimum if

$$a^2 \ge \max\Big\{1, \frac{4^{H+1} m\big(\|Y\|_F^2 - \tilde{\mathcal{L}}_H\big)}{\sigma_{\mathrm{small}}^2\big((XS^H)_{*\mathcal{I}}\big)}\Big\}.$$

Remark 3.4. The convergence rate of the gradient flow explicitly depends on the feature matrix X, graph shift matrix S and the initial weight matrices. The notation $\sigma_{\mathrm{small}}^2((XS^H)_{*\mathcal{I}})$ denotes the smallest non-zero singular value of $(XS^H)_{*\mathcal{I}}$, which implies that the training loss for the semi-supervised learning of graph neural network converges to the global minimum with respect to aggregate matrix S even if $(XS^H)_{*\mathcal{I}}$ has the rank deficiency.

4 Energy optimization in linear GNNs

In this section, we discuss the solution to the global minimum of the mean square loss of the linear GNN that minimizes the total energy of the weight matrices.

Let $G = (V, E)$ be a graph embedded with a feature matrix $X \in \mathbb{R}^{d_x \times n}$, and a labeled matrix $Y \in \mathbb{R}^{d_y \times \bar{n}}$. The linear GNN with H layers and weight matrices $\boldsymbol{W}$ is defined by (2.1). In general, the global minimum of the mean square loss $\mathcal{L}(\boldsymbol{W})$ of the linear GNN satisfy

$$\min_{\boldsymbol{W}} \mathcal{L}(\boldsymbol{W}) \geq \min_{W \in \mathbb{R}^{d_y \times d_x}} \frac{1}{\bar{n} d_y} \big\| W(XS^H)_{*\mathcal{I}} - Y \big\|_F^2.$$

The equality holds if for each $1 \leq \ell \leq H$, the hidden feature dimension $d_\ell \geq \min\{d_x, d_y\}$.

Let $W \in \mathbb{R}^{d_y \times d_x}$ be arbitrary, and let the rank of $(XS^H)_{*\mathcal{I}} \in \mathbb{R}^{d_x \times \bar{n}}$ be k. Then, from the singular value decomposition of $(XS^H)_{*\mathcal{I}}$ we have

$$\begin{aligned} \big\| W(XS^H)_{*\mathcal{I}} - Y \big\|_F^2 =& \big\| WU\Sigma V^\top - Y \big\|_F^2 = \| WU\Sigma - YV \|_F^2 \\ =& \sum_{i=1}^{d_y} \sum_{j=1}^{k} \Big(\sigma_j\big((XS^H)_{*\mathcal{I}}\big)[WU]_{ij} - [YV]_{ij} \Big)^2 + \sum_{i=1}^{d_y} \sum_{j=k+1}^{\bar{n}} [YV]_{ij}^2, \end{aligned}$$

where $[P]_{ij}$ denotes the i, j-th entry of the matrix P. Hence, the global minimum of least square equation $\|W(XS^H)_{*\mathcal{I}} - Y\|_F^2$ is attained for all $W \in \mathbb{R}^{d_y \times d_x}$ such that

$$\big[WU\big]_{ij} = \frac{1}{\sigma_j((XS^H)_{*\mathcal{I}})} [YV]_{ij}, \qquad \text{for all } 1 \leq i \leq d_y,\ 1 \leq j \leq k.$$

In particular, the global minimum solution W also minimizes the total energy $\|W\|_F$ if and only if $[WU]_{ij} = 0$ for $1 \leq i \leq d_y$, $k+1 \leq j \leq \bar{n}$. Hence, $\tilde{W}$ is the unique solution to the global minimum of $\mathcal{L}$ with minimum energy and $\tilde{W}$ is given by

$$\tilde{W} = Y(XS^H)^\dagger_{*\mathcal{I}}.$$

The problem of identifying the solution to the minimization of square loss with minimum energy of the weights for general linear GNN is non-trivial. Let the collection $\tilde{\boldsymbol{W}} = \big(\tilde{W}_1, \tilde{W}_2, \ldots, \tilde{W}_{H+1}\big)$ be the solution to the global minimum of the square loss. Then $\tilde{\boldsymbol{W}}$ minimizes the energy of the product $\tilde{W}_{H+1}\tilde{W}_H \cdots \tilde{W}_1$ if and only if

$$\tilde{W}_{H+1}\tilde{W}_H \cdots \tilde{W}_1 = Y(XS^H)^\dagger_{*\mathcal{I}}. \tag{4.1}$$

However, for $H \geq 1$, the collection $\tilde{\boldsymbol{W}}$ might not minimize the energy of the individual weights matrices.

In the following theorem, we address the optimization problem for the linear GNNs with H layers defined as follows.

$$\begin{aligned} &\min_{\tilde{W}} \sum_{\ell=1}^{H+1} \|\tilde{W}_\ell\|_F^2 \\ &\text{subject to } \tilde{W}_{H+1}\tilde{W}_H \cdots \tilde{W}_1 = Y(XS^H)^\dagger_{*\mathcal{I}}. \end{aligned} \tag{4.2}$$

Before answering the above optimization problem, first we define the *balanced* condition on the collections of weight matrices.

Definition 4.1. The collection $\boldsymbol{W} = \big(W_1, W_2, \ldots, W_H\big)$ is said to be *balanced* if for every $1 \leq \ell \leq H$,

$$W_\ell W_\ell^\top = W_{\ell+1}^\top W_{\ell+1}.$$

Theorem 4.2. *Let $G = (V, E)$ be a graph embedded with a feature matrix $X \in \mathbb{R}^{d_x \times n}$, and a labeled matrix $Y \in \mathbb{R}^{d_y \times \bar{n}}$. Let us consider a linear GNN with H layers and S be a graph shift matrix. If the collection $\boldsymbol{W}$ is balanced and satisfy the equation (4.1), then $\boldsymbol{W}$ is the solution to the optimization problem (4.2).*

Proof. Let $\boldsymbol{W}$ be the collection of weight matrices of the linear GNN with H layers. In order to solve the optimization problem, we use the method of Lagrange multiplier and the Lagrangian function is defined by

$$\mathcal{F}(\boldsymbol{W}) = \sum_{\ell=1}^{H+1} \|W_\ell\|_F^2 + \operatorname{trace}\big(\Lambda^\top (W_{H+1} W_H \cdots W_1 - Y(XS^H)^\dagger_{*\mathcal{I}})\big),$$

where $\Lambda \in \mathbb{R}^{d_y \times d_x}$ is the Lagrange multiplier. All the critical points of $\mathcal{F}$ is given by

$$\frac{\partial \mathcal{F}}{\partial \Lambda} = 0, \qquad \text{and} \qquad \frac{\partial \mathcal{F}}{\partial W_\ell} = 0, \quad \text{for } 1 \le \ell \le H+1. \tag{4.3}$$

The fact $\frac{\partial}{\partial Q}\operatorname{trace}(PQR) = P^\top R^\top$ for any $P \in \mathbb{R}^{d \times d'}$, $Q \in \mathbb{R}^{d' \times d''}$ and $R \in \mathbb{R}^{d'' \times d'''}$, and the equation (4.3) implies that the critical points of $\mathcal{F}$ is balanced and satisfy (4.1).

Let $\boldsymbol{W}$ be a critical points of $\mathcal{F}$. From the balanced condition of $\boldsymbol{W}$ and (4.1), one can easily verify that

$$\begin{aligned}
\big(Y(XS^H)^\dagger_{*\mathcal{I}}\big)^\top \big(Y(XS^H)^\dagger_{*\mathcal{I}}\big) &= \big(W_1^\top W_1\big)^{H+1}, \\
\big(Y(XS^H)^\dagger_{*\mathcal{I}}\big) \big(Y(XS^H)^\dagger_{*\mathcal{I}}\big)^\top &= \big(W_{H+1} W_{H+1}^\top\big)^{H+1}.
\end{aligned}$$

Hence, the eigenspace of $W_1^\top W_1$ and $\big(Y(XS^H)^\dagger_{*\mathcal{I}}\big)^\top \big(Y(XS^H)^\dagger_{*\mathcal{I}}\big)$ are identical. In particular, if λ is the eigenvalue of $\big(Y(XS^H)^\dagger_{*\mathcal{I}}\big)^\top \big(Y(XS^H)^\dagger_{*\mathcal{I}}\big)$ then $\lambda^{\frac{1}{H+1}}$ is the eigenvalue of $W_1^\top W_1$. Hence, the minimum value of the total energy of weight matrices for the optimization problem (4.2) is given by

$$\sum_{\ell=1}^{H+1} \|W_\ell\|_F^2 = (H+1)\|W_1\|_F^2 = (H+1) \sum_{i=1}^{\min\{d_x, d_y\}} \sigma_i^{\frac{2}{H+1}} \big(Y(XS^H)^\dagger_{*\mathcal{I}}\big).$$

This completes the proof. □

Remark 4.3. The balancedness condition of the collection $\boldsymbol{W}$ provide algebraic relation between weight matrices. For example, suppose σ is a non-zero singular value of W_1, then the relation $W_1 W_1^\top = W_2^\top W_2$ implies σ is also a singular value of W_2. In particular, σ is the non-zero singular value of W_ℓ for each $1 \le \ell \le H+1$. Hence, all weight matrices W_ℓ share the same set of non-zero singular values.

The global minimum of (2.2) with balanced condition, minimizes the total energy of the weights. This initiates another non-trivial challenge, whether the initialization leads the gradient flow to the solution of (4.2). The articles [3, 6, 14] studied closely into this direction for linear deep neural networks with approximately balanced initialization. However, they did not discuss on the optimization problem (4.2) and studied the convergence of the gradient dynamics with full rank assumption on the input data. Even though the convergence analysis discussed in Section 3 provides a sufficient initialization for the convergent of the square loss to the global minimum, but it might not optimize the total energy of the weight matrices. In the following theorem, we answered this question with some weaker assumption on the hidden dimension.

Theorem 4.4. *Let $G = (V, E)$ be a graph embedded with a feature matrix $X \in \mathbb{R}^{d_x \times n}$, and a labeled matrix $Y \in \mathbb{R}^{d_y \times \bar{n}}$ with rank $\big(Y(XS^H)^\top_{*\mathcal{I}}\big) = \underline{k}$. Let us consider a linear GNN with H layers such that $d_y \leq \min\{d_1, d_2, \ldots, d_H\}$ and S as aggregation matrix. The gradient flow started at $\boldsymbol{W}(0)$ converges to the solution of the optimization problem (4.2) if the following holds.*

1. *$\boldsymbol{W}(0)$ is balanced.*
2. *If rank $\big((XS^H)_{*\mathcal{I}}\big) = k$ and the orthonormal matrix $U \in \mathbb{R}^{d_x \times d_x}$ is the collection of left singular vector of $(XS^H)_{*\mathcal{I}}$, then for all $1 \leq i \leq d_1$, $k < j \leq d_x$, $\big[W_1(0)U\big]_{ij} = 0$,*
3. *Rank of $W_1(0)$ is $\underline{k}$ with $\underline{k}$-th singular value satisfy*

$$L\big(\boldsymbol{W}(0)\big) \leq \frac{1}{4^{H+1}m}\sigma_{\underline{k}}^{2H}(W_1(0))\sigma^2_{\text{small}}\big((XS^H)_{*\mathcal{I}}\big) \tag{4.4}$$

where $\underline{k} = \min\{d_y, k\}$, $m = \bar{n}d_x$, and L is the loss function defined by (3.3).

Proof. In linear deep neural network with squared loss, the balanced relation is preserved by the gradient flow equation, see [3, 6, 14]. In the case of linear graph neural networks, it is easy to check that for $1 \leq \ell \leq H+1$

$$\frac{dW_\ell}{dt} = -\frac{2}{\bar{n}d_y}\big(W_{H+1}W_H \cdots W_{\ell+1}\big)^\top(\hat{Y} - Y)(XS^H)^\top_{*\mathcal{I}}\big(W_{\ell-1}W_{\ell-2}\cdots W_1\big)^\top.$$

Hence, we have

$$\frac{d}{dt}\big(W_\ell W_\ell^\top\big) = \frac{d}{dt}\big(W_{\ell+1}^\top W_{\ell+1}\big), \qquad \forall 1 \leq \ell \leq H.$$

If the weight parameters are initially balanced then $\boldsymbol{W}(t)$ is balanced for each $t \geq 0$.

Let us assume that the gradient flow converges to the global minimum at $\tilde{\boldsymbol{W}}$. The global minimizer $\tilde{\boldsymbol{W}}$ satisfy (4.1) if and only if

$$\big[\tilde{W}_{H+1}\tilde{W}_H \cdots \tilde{W}_1 U\big]_{ij} = 0 \qquad \text{for all } 1 \leq i \leq d_y,\ k < j \leq d_x.$$

Hence it is enough to prove that at any time $t \geq 0$, $\boldsymbol{W}(t)$ satisfy

$$\big[W_{H+1}W_H \cdots W_1 U\big]_{ij}(t) = 0 \qquad \text{for all } 1 \leq i \leq d_y,\ k < j \leq d_x.$$

First, we observe that for $\ell = 1$ the gradient flow equation implies

$$\begin{aligned}\frac{d(W_1U)}{dt} &= -\nabla_{W_1}\mathcal{L}(\boldsymbol{W})U \\ &= -\frac{2}{m}\big(W_{H+1}W_H \cdots W_2\big)^\top\big(W_{H+1}W_H \cdots W_1(XS^H)_{*\mathcal{I}} - Y\big)V\Sigma^\top.\end{aligned}$$

This implies, $\left[\frac{d(W_1U)}{dt}\right]_{ij} = 0$ for $1 \leq i \leq d_1$ and $k < j \leq d_x$, and therefore

$$\big[W_1U\big]_{ij}(t) = \big[W_1U\big]_{ij}(0) = 0, \qquad \forall\, t \geq 0.$$

Suppose there exists $\ell \geq 1$ such that for $1 \leq i \leq d_\ell$, and $k < j \leq d_x$, we have

$$\big[W_\ell W_{\ell-1} \cdots W_1 U\big]_{ij}(t) = 0, \qquad \text{for } t \geq 0.$$

The chain rule for differentiation implies

$$\frac{d}{dt}\big(W_{\ell+1}W_\ell \cdots W_1 U\big) = \frac{dW_{\ell+1}}{dt}W_\ell \cdots W_1 U + W_{\ell+1}\frac{d}{dt}\big(W_\ell \cdots W_1 U\big).$$

Hence, for all $t \geq 0$, $1 \leq i \leq d_{\ell+1}$ and $k < j \leq d_x$

$$\Big[\frac{d}{dt}\big(W_{\ell+1}W_\ell \cdots W_1 U\big)\Big]_{ij} = 0,$$
$$\big[W_{\ell+1}W_\ell \cdots W_1 U\big]_{ij}(t) = 0.$$

Therefore, from mathematical induction we have

$$\big[W_{H+1}W_H \cdots W_1 U\big]_{ij}(t) = 0, \qquad \forall\, t \geq 0,\, 1 \leq i \leq d_y,\, k < j \leq d_x.$$

Finally, we now ensure convergence of the gradient flow training of linear GNN with balanced initialization and some minor assumption on $W_1(0)$. In order to study the convergence result, we consider the following two cases.

First assume that $d_y \leq \operatorname{rank}\big((XS^H)_{*\mathcal{I}}\big)$. The equation (3.4) implies

$$\begin{aligned}\|\nabla_{\operatorname{vec}(W_1)} L(\boldsymbol{W})\|_2^2 &\geq \frac{4}{m^2}\sigma_{d_y}^2\big(W_2^\top W_3^\top \cdots W_{H+1}^\top\big)\big\|(\hat{Y}-Y)(XS^H)_{*\mathcal{I}}^\top\big\|_F^2 \\ &= \frac{4}{m^2}\sigma_{d_y}^{2H}\big(W_{H+1}\big)\big\|(\hat{Y}-Y)(XS^H)_{*\mathcal{I}}^\top\big\|_F^2.\end{aligned}$$

The last equality follows from the balancedness of $\boldsymbol{W}(t)$ for each $t \geq 0$.

Secondly, assume that $k := \operatorname{rank}\big((XS^H)_{*\mathcal{I}}\big) \leq d_y$ and $(XS^H)_{*\mathcal{I}} = U\Sigma V^\top$ is the singular value decomposition of $(XS^H)_{*\mathcal{I}}$. Then the gradient of L with respect to $\operatorname{vec}\big(W_{H+1}\big)$ implies

$$\begin{aligned}\|\nabla_{\operatorname{vec}(W_{H+1})} L(\boldsymbol{W})\|_2^2 &= \frac{4}{m^2}\big\|(\hat{Y}-Y)V\Sigma^\top U^\top W_1^\top W_2^\top \cdots W_H^\top\big\|_F^2 \\ &= \frac{4}{m^2}\big\|W_H W_{H-1}\cdots W_1 U\Sigma V^\top(\hat{Y}-Y)^\top\big\|_F^2.\end{aligned}$$

The entries of j-th column with $j > k$ in the matrix $W_H W_{H-1}\cdots W_1 U \in \mathbb{R}^{d_H \times d_x}$ are zero. Simultaneously, the entries of i-th row with $i > k$ in the matrix $\Sigma V^\top(\hat{Y}-Y)^\top$ are zero. Hence, we get

$$\begin{aligned}\|\nabla_{\operatorname{vec}(W_{H+1})} L(\boldsymbol{W})\|_2^2 &= \frac{4}{m^2}\big\|W\bar{Y}\big\|_F^2 \geq \frac{4}{m^2}\sigma_{\min}^2(W)\|\bar{Y}\|_F^2 \\ &= \frac{4}{m^2}\sigma_k^2(W_H W_{H-1}\cdots W_1 U)\|(\hat{Y}-Y)V\Sigma^\top\|_F^2 \\ &= \frac{4}{m^2}\sigma_k^{2H}(W_1)\big\|(\hat{Y}-Y)(XS^H)_{*\mathcal{I}}^\top\big\|_F^2.\end{aligned}$$

In the first equality $W \in \mathbb{R}^{d_H \times k}$ and $\bar{Y} \in \mathbb{R}^{k \times d_y}$ are the matrix with first k columns of $W_H W_{H-1}\cdots W_1 U$ and first k rows of $\Sigma V^\top(\hat{Y}-Y)^\top$ respectively.

If the $\boldsymbol{W}(0)$ is balanced with $\operatorname{rank}(W_\ell(0)) = \underline{\mathrm{k}}$ and j-th column of $W_1(0)U$ is a zero vector for each $j > \operatorname{rank}\big((XS^H)_{*\mathcal{I}}\big)$, then each $\boldsymbol{W}_\ell(t)$ in the collection $\boldsymbol{W}(t)$ shares the set of non-zero singular values for $t \geq 0$. Therefore, following the similar arguments as in Theorem 3.1, the gradient flow converges to the global minimum provided the $\underline{\mathrm{k}}$-th singular value of $W_\ell(0)$ satisfy

$$L(\boldsymbol{W}(0)) \leq \frac{1}{4^{H+1}m}\sigma_{\underline{\mathrm{k}}}^{2H}(W_\ell(0))\sigma_{\text{small}}^2\big((XS^H)_{*\mathcal{I}}\big).$$

This completes the proof. □

The assumption (4.4) on the initial weights is equivalent to the inequality (3.8) for balanced initialization. The $\underline{\mathrm{k}}$-th singular value of the weights at time $t \geq 0$ in the gradient flow training always greater than $\sigma_{\underline{\mathrm{k}}}(W_\ell(0))/2$ and the weights converges to the solution of optimization problem (4.2). Therefore, the balanced initialization satisfy

$$\sigma_{\underline{\mathrm{k}}}(W_\ell(0)) \leq 2\,\sigma_{\underline{\mathrm{k}}}^{\frac{1}{H+1}}\big(Y(XS^H)_{*\mathcal{I}}\big).$$

The balanced condition imposes such restriction on the initialization which causes the inequality (4.4) infeasible.

5 Convergence analysis of gradient descent training

In this section, we study the gradient descent training of linear GNNs and discuss its convergence. The gradient descent training is an analogue of gradient flow training of GNNs. The weight parameters are updated in the direction of the negative gradient with infinitesimally small steps in a gradient flow training, however, stepsize is fixed in the gradient descent training. Therefore, with an inappropriate stepsize the weight parameter diverges to infinity in gradient descent method.

The following theorem is the analogue of Theorem 3.1 for gradient descent. The idea of the proof is to find an appropriate stepsize such that the weights in each iterations are lies in the neighborhood of the initial weights. The loss $\mathcal{L}$ corresponds to the weight sequences is a decreasing sequence bounded below by global minimum.

Theorem 5.1. *Let $G = (V, E)$ be a graph embedded with a feature matrix $X \in \mathbb{R}^{d_x \times n}$, and a labeled matrix $Y \in \mathbb{R}^{d_y \times \bar{n}}$. Let us consider a linear GNN with H layers and S be the graph shift matrix, and consider the mean square loss $\mathcal{L}$ define by (2.2). With a sufficiently small stepsize $\eta > 0$ and the initial weights considered in Theorem 3.1, the gradient descent*

$$W_\ell^{(k+1)} = W_\ell^{(k)} - \eta \nabla_{W_\ell} \mathcal{L}\big(\boldsymbol{W}^{(k)}\big), \qquad \text{for } 1 \le \ell \le H+1, \tag{5.1}$$

converges to global minimum of $\mathcal{L}$ as $k \to \infty$. Here $\boldsymbol{W}^{(0)} = \boldsymbol{W}(0)$.

Moreover, for $k \ge 0$,

$$\mathcal{L}(\boldsymbol{W}^{(k)}) - \tilde{\mathcal{L}}_H \le \left(1 - \frac{\beta \sigma^2_{\text{small}}\big((XS^H)_{*\mathcal{I}}\big)\eta}{2m}\right)^k \big(\mathcal{L}(\boldsymbol{W}^{(0)}) - \tilde{\mathcal{L}}_H\big).$$

Proof. For $1 \le \ell \le H+1$, the gradient descent algorithm (5.1) can be written as follows:

$$\operatorname{vec}\big(\boldsymbol{W}^{(k+1)}\big) = \operatorname{vec}\big(\boldsymbol{W}^{(k)}\big) - \eta \nabla \mathcal{L}\big(\operatorname{vec}\big(\boldsymbol{W}^{(k)}\big)\big), \tag{5.2}$$

where $\operatorname{vec}\big(\boldsymbol{W}^{(k)}\big)$ is a vector form of the collection $\big(W_1^{(k)}, W_2^{(k)}, \dots, W_{H+1}^{(k)}\big)$.

The loss function $\mathcal{L}$ is defined by (2.2), is twice continuously differentiable. Hence, there exists a positive constant M such that for each $\operatorname{vec}(\boldsymbol{W})$ in a closed bounded ball $B_{2r}\big(\operatorname{vec}\big(\boldsymbol{W}^{(0)}\big)\big) \subset \mathbb{R}^p$, absolute values of the entries in the gradient and the Hessian matrix of $\mathcal{L}$ is bounded by M, i.e.,

$$\Big|\frac{\partial \mathcal{L}(\operatorname{vec}\big(\boldsymbol{W}\big))}{\partial \operatorname{vec}\big(\boldsymbol{W}\big)_j}\Big| \le M \qquad \text{and} \qquad \Big|\frac{\partial^2 \mathcal{L}(\operatorname{vec}\big(\boldsymbol{W}\big))}{\partial \operatorname{vec}\big(\boldsymbol{W}\big)_i \partial \operatorname{vec}\big(\boldsymbol{W}\big)_j}\Big| \le M$$

where r and p are the same as define in Theorem 3.1.

Let $k \ge 1$ be such that $\operatorname{vec}\big(\boldsymbol{W}^{(j)}\big) \in B_r\big(\operatorname{vec}\big(\boldsymbol{W}^{(0)}\big)\big)$ for each $1 \le j \le k$. Claim that for sufficiently small η, $\operatorname{vec}\big(\boldsymbol{W}^{(k+1)}\big) \in B_r\big(\operatorname{vec}\big(\boldsymbol{W}^{(0)}\big)\big)$.

For $\eta < \frac{r}{M\sqrt{p}}$ and (5.2) implies

$$\big\|\operatorname{vec}\big(\boldsymbol{W}^{(j+1)}\big) - \operatorname{vec}\big(\boldsymbol{W}^{(j)}\big)\big\|_2 = \eta \big\|\nabla \mathcal{L}\big(\operatorname{vec}\big(\boldsymbol{W}^{(j)}\big)\big)\big\|_2 \le \eta M \sqrt{p} < r.$$

Hence, $\operatorname{vec}\big(\boldsymbol{W}^{(j+1)}\big) \in B_{2r}\big(\operatorname{vec}\big(\boldsymbol{W}^{(0)}\big)\big)$, and the line segment joining $W_\ell^{(j+1)}$ and $W_\ell^{(j)}$ is in $B_{2r}\big(\operatorname{vec}\big(\boldsymbol{W}^{(0)}\big)\big)$.

Let $R_j := \mathcal{L}\big(\operatorname{vec}\big(\boldsymbol{W}^{(j+1)}\big)\big) - \mathcal{L}\big(\operatorname{vec}\big(\boldsymbol{W}^{(j)}\big)\big) + \eta \|\nabla \mathcal{L}\big(\operatorname{vec}\big(\boldsymbol{W}^{(j)}\big)\big)\|_2^2$. The second order expansion of $\mathcal{L}$ implies

$$\begin{aligned} R_j &= \frac{1}{2}\eta^2 \nabla \mathcal{L}\big(\operatorname{vec}\big(\boldsymbol{W}^{(j)}\big)\big) \cdot \nabla^2 \mathcal{L}\big(\operatorname{vec}\big(\boldsymbol{W}^*\big)\big) \nabla \mathcal{L}\big(\operatorname{vec}\big(\boldsymbol{W}^{(j)}\big)\big) \\ &\le \frac{1}{2}\eta^2 M p \|\nabla \mathcal{L}(\operatorname{vec}\big(\boldsymbol{W}^{(j)}\big))\|_2^2 \\ &\le \frac{1}{2}\eta \|\nabla \mathcal{L}(\operatorname{vec}\big(\boldsymbol{W}^{(j)}\big))\|_2^2 \end{aligned}$$

where $\mathrm{vec}(\boldsymbol{W}^*)$ is some point in the line segment joining $\mathrm{vec}(\boldsymbol{W}^{(j+1)})$ and $\mathrm{vec}(\boldsymbol{W}^{(j)})$, and $\eta < \min\{\frac{1}{Mp}, \frac{r}{M\sqrt{p}}\}$.

Hence, the training loss $\mathcal{L}$ decreases at each iterations as

$$\frac{1}{2}\eta\big\|\nabla\mathcal{L}\big(\mathrm{vec}(\boldsymbol{W}^{(j)})\big)\big\|_2^2 \le \mathcal{L}\big(\mathrm{vec}(\boldsymbol{W}^{(j)})\big) - \mathcal{L}\big(\mathrm{vec}(\boldsymbol{W}^{(j+1)})\big). \tag{5.3}$$

Moreover, for $1 \le j \le k$, we have

$$\begin{aligned}
\mathcal{L}(\mathrm{vec}(\boldsymbol{W}^{(j+1)})) &= \mathcal{L}(\mathrm{vec}(\boldsymbol{W}^{(j)})) - \eta\|\nabla\mathcal{L}(\mathrm{vec}(\boldsymbol{W}^{(j)}))\|_2^2 + R_j \\
&\le \mathcal{L}(\mathrm{vec}(\boldsymbol{W}^{(j)})) - \frac{1}{2}\eta\|\nabla\mathcal{L}(\mathrm{vec}(\boldsymbol{W}^{(j)}))\|_2^2 \\
\Big(\mathcal{L}\big(\mathrm{vec}(\boldsymbol{W}^{(j+1)})\big) - \tilde{\mathcal{L}}_H\Big) &\le \Big(1 - \frac{\eta\alpha}{2}\Big)\Big(\mathcal{L}\big(\mathrm{vec}(\boldsymbol{W}^{(j)})\big) - \tilde{\mathcal{L}}_H\Big) \\
\Big(\mathcal{L}\big(\mathrm{vec}(\boldsymbol{W}^{(j+1)})\big) - \tilde{\mathcal{L}}_H\Big) &\le \Big(1 - \frac{\eta\alpha}{2}\Big)^{j+1}\Big(\mathcal{L}\big(\mathrm{vec}(\boldsymbol{W}^{(0)})\big) - \tilde{\mathcal{L}}_H\Big)
\end{aligned} \tag{5.4}$$

where $\alpha := \inf\Big\{\frac{|\nabla\mathcal{L}(\boldsymbol{W})|^2}{\mathcal{L}(\boldsymbol{W})-\tilde{\mathcal{L}}_H} : \mathrm{vec}(\boldsymbol{W}) \in B_r\big(\mathrm{vec}(\boldsymbol{W}^{(0)})\big)\Big\}$.

For $1 \le j \le k$, and the gradient descent algorithm (5.2) implies

$$\big\|\mathrm{vec}(\boldsymbol{W}^{(k+1)}) - \mathrm{vec}(\boldsymbol{W}^{(j)})\big\|_2 \le \sum_{i=j}^{k}\eta\big\|\nabla\mathcal{L}\big(\mathrm{vec}(\boldsymbol{W}^{(i)})\big)\big\|_2. \tag{5.5}$$

Let L be the relative loss of $\mathcal{L}$ with respect to global minimum, i.e.,

$$L\big(\mathrm{vec}(\boldsymbol{W})\big) := \mathcal{L}(\mathrm{vec}(W)) - \tilde{\mathcal{L}}_H.$$

The upper bound of the sum of the gradient is estimated by (5.3) and (5.4).

$$\begin{aligned}
\sum_{i=j}^{k}\eta\big\|\nabla\mathcal{L}\big(\mathrm{vec}(\boldsymbol{W}^{(i)})\big)\big\|_2 &\le \sum_{i=j}^{k}\sqrt{2\eta\Big(L\big(\mathrm{vec}(\boldsymbol{W}^{(i)})\big) - L\big(\mathrm{vec}(\boldsymbol{W}^{(i+1)})\big)\Big)} \\
&= \sqrt{2\eta}\sum_{i=j}^{k}\Big(\sqrt{L\big(\mathrm{vec}(\boldsymbol{W}^{(i)})\big)} + \sqrt{L\big(\mathrm{vec}(\boldsymbol{W}^{(i+1)})\big)}\Big)^{\frac{1}{2}}\times \\
&\qquad\qquad \Big(\sqrt{L\big(\mathrm{vec}(\boldsymbol{W}^{(i)})\big)} - \sqrt{L\big(\mathrm{vec}(\boldsymbol{W}^{(i+1)})\big)}\Big)^{\frac{1}{2}} \\
&\le \sqrt{4\eta}\Big(L\big(\mathrm{vec}(\boldsymbol{W}^{(j)})\big)\Big)^{\frac{1}{4}}\Big(\sum_{i=j}^{k}\sqrt{L\big(\mathrm{vec}(\boldsymbol{W}^{(i)})\big)}\Big)^{\frac{1}{2}} \\
&\le \Big(4\eta L\big(\mathrm{vec}(\boldsymbol{W}^{(0)})\big)\Big)^{\frac{1}{2}}\Big((1-\frac{\eta\alpha}{2})^{\frac{j}{2}}\sum_{i=j}^{k}(1-\frac{\eta\alpha}{2})^{\frac{i}{2}}\Big)^{\frac{1}{2}} \\
&\le \Big(4\eta L\big(\boldsymbol{W}^{(0)}\big)\Big)^{\frac{1}{2}}\Big((1-\frac{\eta\alpha}{2})^{j}\sum_{i=0}^{k-j}(1-\frac{\eta\alpha}{2})^{\frac{i}{2}}\Big)^{\frac{1}{2}} \\
&\le (1-\frac{\eta\alpha}{2})^{\frac{j}{2}}\Big(4\eta L\big(\boldsymbol{W}^{(0)}\big)\Big)^{\frac{1}{2}}\Big(\sum_{i=0}^{k-j}(1-\frac{\eta\alpha}{4})^{i}\Big)^{\frac{1}{2}} \\
&\le \Big(1-\frac{\eta\alpha}{2}\Big)^{\frac{j}{2}}\Big(\frac{16L(\boldsymbol{W}^{(0)})}{\alpha}\Big)^{\frac{1}{2}}
\end{aligned}$$

The inequality (3.7) implies α is bounded below by $\frac{1}{m}\beta\sigma_{\mathrm{small}}\big((XS^H)_{*\mathcal{I}}\big)$.
Let $\eta < \min\Big\{\frac{r}{M\sqrt{p(H+1)}}, \frac{1}{Mp}, \frac{2m}{\beta\sigma^2_{\mathrm{small}}((XS^H)_{*\mathcal{I}})}\Big\}$, then the inequality (5.5) implies

$$\big\|\mathrm{vec}(\boldsymbol{W}^{(k+1)}) - \mathrm{vec}(\boldsymbol{W}^{(j)})\big\|_2 \le \Big(1 - \frac{\beta\sigma_{\mathrm{small}}\big((XS^H)_{*\mathcal{I}}\big)\eta}{2m}\Big)^{\frac{j}{2}}\Big(\frac{16mL\big(\boldsymbol{W}^{(0)}\big)}{\beta\sigma_{\mathrm{small}}\big((XS^H)_{*\mathcal{I}}\big)}\Big)^{\frac{1}{2}}.$$

Moreover, if the initial choice of $\boldsymbol{W}(0)$ with $\sigma_{\min}(W_{H+1}(0))$ is sufficiently large such that

$$\frac{16L(\boldsymbol{W}^{(0)})}{r^2} < \frac{1}{m}\beta\sigma^2_{\text{small}}\big((XS^H)_{*\mathcal{I}}\big),$$

then $\text{vec}\big(\boldsymbol{W}^{(k)}\big) \in B_r\big(\text{vec}\big(\boldsymbol{W}^{(0)}\big)\big)$. Hence, by mathematical induction

$$\text{vec}\big(\boldsymbol{W}^{(k)}\big) \in B_r\big(\text{vec}\big(\boldsymbol{W}^{(0)}\big)\big) \qquad \forall\, k \geq 0.$$

The sequence of weights $\{\text{vec}\big(\boldsymbol{W}^{(k)}\big)\}_{k\geq 0}$ is a bounded contraction sequence, hence, a Cauchy sequence. In particular, for $\varepsilon > 0$ there exists $M > 0$ such that

$$\big\|\text{vec}\big(\boldsymbol{W}^{(k)}\big) - \text{vec}\big(\boldsymbol{W}^{(j)}\big)\big\|_2 < \varepsilon, \qquad k, j > M,$$

where $M = 2\log\Big(\frac{r}{\varepsilon\sqrt{H+1}}\Big)/\log\Big(\frac{2m}{2m-\beta\sigma^2_{\text{small}}((XS^H)_{*\mathcal{I}})\eta}\Big)$. This implies, weights converges to some $\text{vec}\big(\tilde{\boldsymbol{W}}\big) \in B_r\big(\text{vec}\big(\boldsymbol{W}^{(0)}\big)\big)$.

Moreover, the inequality (5.4) implies

$$\mathcal{L}(\boldsymbol{W}^{(k)}) - \tilde{\mathcal{L}}_H \leq \Big(1 - \frac{\beta\sigma^2_{\text{small}}((XS^H)_{*\mathcal{I}})\eta}{2m}\Big)^k\big(\mathcal{L}(\boldsymbol{W}^{(0)}) - \tilde{\mathcal{L}}_H\big).$$

This completes the proof. □

Remark 5.2. Let ε be any arbitrary small positive real. Theorem 5.1 shows that loss $\mathcal{L}$ converges to the global minimum $\tilde{\mathcal{L}}_H$ exponentially as the iteration progresses. In particular, $\mathcal{L}$ is close to $\tilde{\mathcal{L}}_H$ with ε perturbation, i.e., $\mathcal{L}(\boldsymbol{W}^{(k)}) - \tilde{\mathcal{L}}_H < \varepsilon$, if

$$k > \Big(\log\frac{\mathcal{L}(\boldsymbol{W}^{(0)}) - \tilde{\mathcal{L}}_H}{\varepsilon}\Big)/\Big(\log\frac{2m}{2m - \beta\sigma^2_{\text{small}}((XS^H)_{*\mathcal{I}})\eta}\Big).$$

6 Numerical experiments

In this section, we illustrate aspects of the gradient flow dynamics such as the dependence of the convergence rate on the graph topology and the choice of the shift operator S in terms of the principle singular value $\sigma_{\text{small}}\big((XS^H)_{*\mathcal{I}}\big)$. We provide a set of numerical experiments on the synthetic data following well-established undirected graph models (Erdős–Rényi, k-nearest neighbors, SBM, Barabási-Albert) and the real-world dataset of CDC climate data in US counties [1].

6.1 Synthetic data

Let us briefly describe synthetic graph models we use to illustrate the convergence of the linear GNN (see corresponding examples in Figure 1):

(i) **Erdős–Rényi graph,** $G(n,p)$: A random graph on n nodes where the edge $\{i,j\}$ enters the graph with probability p independently on all the other edges. The probability p naturally defines the connectivity of the generated graph; we assume $p > (1+\varepsilon)\ln n/n$ such that the generated graph is almost surely connected, [19].

(ii) **k-nearest neighbors graph,** $K_k(n)$: A regular, highly-symmetric deterministic graph on n nodes. Assume all n nodes to be uniformly and equidistantly placed onto a unit circle; then each nodes is set to be connected by an edge to its k nearest neighbors (in Eucledean distance). For simplicity, we assume k to be even.

(iii) **Stochastic Block Model (SBM)**: A structured generalization of $G(n,p)$ model where n nodes are divided into r classes (blocks) with the probability matrix $P \in \mathbb{R}^{r\times r}$ containing inter- and intra-classes edge probabilities, [28]. In the scope of the current work we assume two block setup $\{n_1, n_2\}$ (such that $n = n_1 + n_2$) with $P = \begin{pmatrix} p & q \\ q & p \end{pmatrix}$ and $p \gg q$; we refer to such models as $\mathrm{SBM}(n_1, n_2, p, q)$.

(iv) **Barabási-Albert model,** $\mathrm{BA}(n,m)$: A scale-free random graph model on n nodes where the node degree distribution follows the power law k^γ with $2 < \gamma < 3$, [7]. Parameter m correspond to the degree of the nodes added to the core graph during generation and typically regulates the minimal node degree.

The feature data $X \in \mathbb{R}^{d_x\times n}$ is assumed to standard normal i.i.d., $x_{ij} \sim \mathcal{N}(0,1)$, for the sake of simplicity. The output label data Y is set to be one-dimensional, $d_y = 1$, with $y_i = f(x_i) + \epsilon \sum_{j:\{i,j\}\in E} g(x_j)$ where $f, g : \mathbb{R}^{d_x} \mapsto \mathbb{R}^{d_y}$ and the value of ϵ controls the impact of the graph structure on the output label. We set $f(x_i) = g(x_i) = \sum x_{ij}$ and $\epsilon = 0.1$ for experiments below and assume that the gradient flow is trained with respect to the mean square error (MSE).

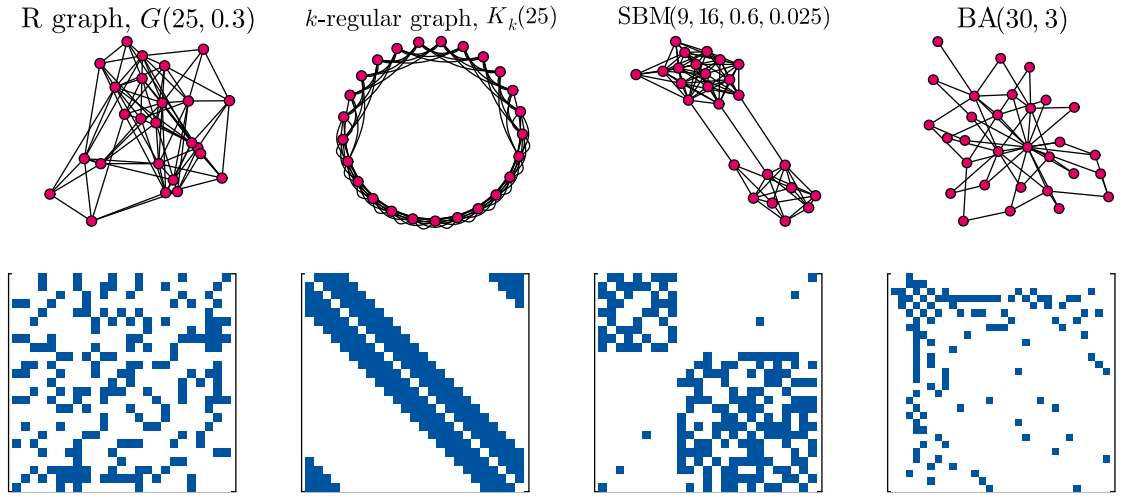


Figure 1: Examples of graphs generated by $G(n,p)$, $K_k(n)$, $\mathrm{SBM}(n_1, n_2, p, q)$, and $\mathrm{BA}(n,m)$ respectively. Sparsity patterns of shift operators (adjacency matrices) are provided in the bottom row.

6.1.1 Singular value $\sigma_{\mathbf{small}}\left(XS^H\right)_{*\mathcal{I}}$

Given the estimate of Theorem 3.2, the convergence rate is determined by the initialization β (i.e. defined in Remark 3.3) and the singular value $\sigma_{\mathrm{small}}\left((XS^H)_{*\mathcal{I}}\right)$ which encodes the graph structure, the initial feature data, and the labeled set.

In Figure 2, we demonstrate the dependence of the singular value $\sigma_{\mathrm{small}}\left((XS^H)_{*\mathcal{I}}\right)$ on the size of the labeled data set $\bar{n} = |\mathcal{I}|$ for the following choices of the shift operator S:

Adj. :	Adjacency Matrix,
S-L Adj. :	Self-Loop Adjacency Matrix,
Nor. S-L Adj. :	Normalized Self-Loop Adjacency Matrix,
L :	Laplacian Matrix,
Nor. L :	Normalized Laplacian Matrix.

For each fixed $\bar{n}$, we uniformly sample a number of labeled sets $\mathcal{I}$ in order to account for possible specific labeled configurations in the graph topology. Note that all the models mainly maintain the same ordering of the singular values in terms of the shift operator, i.e. graph Laplacian generates the highest $\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right)$, normalized self-loop adjacency operator corresponds to the lower singular value, and so on. Additionally, the estimations for adjacency and self-loop adjacency matrices remain numerically close, whilst the position of the corresponding normalized Laplacian curve depends on the model (except BA model which distinguishes between all shift operators): in contrast to $G(n,p)$ and SBM, k-nearest neighbors graphs tend to not set apart the former three shift operators in terms of $\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right)$.

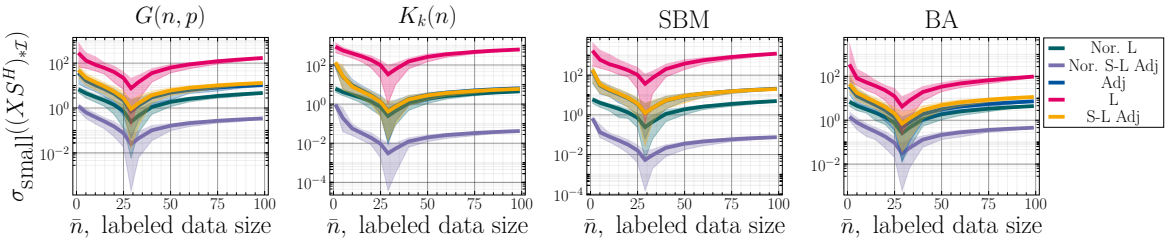


Figure 2: Principle singular value $\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right)$ vs the size of labeled data $\bar{n}$ for various shift operators S in models $G(n,p)$, $K_k(n)$, SBM(n_1,n_2,p,q), and BA(n,m), respectively. Input feature dimension $d_x = 30$, network depth $H = 2$. Solid lines and semi-transparent areas correspond to the average value and spread of $\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right)$ for uniformly sampled sets $\mathcal{I}$.

At the same time, the singular value $\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right)$ for all models and all shift operators exhibits the same universal pattern of behaviour in terms of $\bar{n}$: the monotonicity switch around $\bar{n} = d_x$, for which one can justify with the following probabilistic argument. Let $\Pi_{\mathcal{I}}$ be a binary diagonal matrix such that $[\Pi_{\mathcal{I}}]_{jj} = 1$ if and only if $j \in \mathcal{I}$ and 0 otherwise; then,

$$\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right) = \sigma_{\text{small}}\left(XS^H\Pi_{\mathcal{I}}\right) = \sigma_{\text{small}}\left(\Pi_{\mathcal{I}}S^HX^\top\right) = \min_{\substack{\|z\|=1 \\ z\perp\ker(\Pi_{\mathcal{I}}S^HX^\top)}} \|\Pi_{\mathcal{I}}S^HX^\top z\|.$$

Assume the feature matrix X has a full-row rank, $\operatorname{rank}(X) = d_x < n$, and each $x_i \perp \ker S$ (which is reasonable since any element in the null-space of S is suppressed in the linear GNN (2.1)), then the operator $S^HX^\top$ has a full-column rank, $\operatorname{rank}(S^HX^\top) = d_x$, and a trivial kernel, so $\ker(\Pi_{\mathcal{I}}S^HX^\top) = \ker\Pi_{\mathcal{I}} \cap \operatorname{im}(S^HX^\top)$. Note that $\dim\ker\Pi_{\mathcal{I}} = n - \bar{n}$ and $\dim\operatorname{im}(S^HX^\top) = d_x$, so these two subspaces intersect trivially in general position if $d_x \le \bar{n}$. Hence,

$$\mathbb{E}\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right) = \mathbb{E}\min_{\|z\|=1}\|\Pi_{\mathcal{I}}S^HX^\top z\| = \frac{\bar{n}}{n}\sigma_{\text{small}}(XS^H)$$

where the expectation is taken for uniformly sampled labeled sets $\mathcal{I}$ of a fixed cardinality $\bar{n}$ and $\mathbb{E}\Pi_{\mathcal{I}} = \frac{\bar{n}}{n}I$. Noticeably, each $(d_x \le \bar{n})$-part of the curves on Figure 2 closely follows the average estimation $\frac{\bar{n}}{n}\sigma_{\text{small}}(XS^H)$. At the same time, left parts $d_x > \bar{n}$ correspond to necessarily rank-deficient matrices $(XS^H)_{*\mathcal{I}}$; since the permutation of matrix's columns conserves singular values σ_i and each $(XS^H)_{*\mathcal{I}}$ is a minor of the full matrix XS^H with permuted columns, smaller values of $\bar{n}$ correspond to higher values of $\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right)$ due to the Cauchy's interlace theorem.

Finally, note that in the previous set of assumptions, $\operatorname{rank}(S^H X^\top) = \operatorname{rank}(SX^\top) = d_x$, and $\operatorname{im}(SX^\top)$ is spanned (in probabilistic terms, i.e. randomized matrix sketching) by first d_x dominant singular vectors. Then $\sigma_{\text{small}}(XS^H) = \sigma_{\text{small}}(S^H X^\top) = \sigma_{\text{small}}(S^{H-1}(SX^\top)) = \min_{\|z\|=1} \|S^{H-1}(SX^\top z)\| \approx \lambda_{d_x}^{H-1}\sigma_{\text{small}}(XS)$ where λ_{d_x} denotes the d_x-th largest eigenvalues of the shift operator S. As a result, the value of λ_{d_x} for different shift operators exponentially governs the convergence rate of the training in terms the networks depth H and affects the initialization, Remark 3.3, and the gradient descent step, Theorem 5.1.

6.1.2 Convergence of gradient flow training on the synthetic data

Finally, we provide an illustration of the convergence of the gradient flow training (Theorem 3.1) for all the aforementioned graph models with varying parameters. For each model, graphs on $n = 200$ are considered with 3 most common choices of shift operators: adjacency matrix, Laplacian and normalized Laplacian; input feature data X is sampled normally, $d_x = 50$. We consider a linear GNN architecture with $H = 2$ and hidden dimensions $d_1 = d_2 = 32$ (so $W_1 \in \mathbb{R}^{32\times 50}$, $W_2 \in \mathbb{R}^{32\times 32}$, and $W_3 \in \mathbb{R}^{1\times 32}$); initialization follows Remark 3.3 with $a = 2$. The labeled data set $\mathcal{I}$ is composed of $\bar{n} = 0.75n$ and drawn uniformly; qualitative results below hold for all choices of $\mathcal{I}$. We provide the relative loss $\frac{\mathcal{L}(\boldsymbol{W}(T))-\tilde{\mathcal{L}}_H}{\mathcal{L}(\boldsymbol{W}(0))-\tilde{\mathcal{L}}_H}$ trajectory along the gradient flow and the singular value $\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right)$ for each considered setup; recalling Theorem 3.1, the relative loss is expected to converge exponentially in time.

Along with the exponential convergence of the linear GNN, we demonstrate the following:

- the sparsity of the graph G typically affects the convergence rate with unnormalized shift operators such that sparser systems exhibit slower convergence;
- GNNs using normalized Laplacians tend to be the least affected by the changes in the network structure with $\text{BA}(n, m)$ model showing close-to-none dependency on the injected varying graph structure.

Erdős–Rényi graph, $G(n, p)$. In the classical $G(n, p)$ model, the edge probability p regulates the sparsity pattern (as well as the connectivity and the information spread) of the system; we consider the range of p starting from a sparse but connected graph up to moderate values corresponding to the fast mixing, Figure 3.

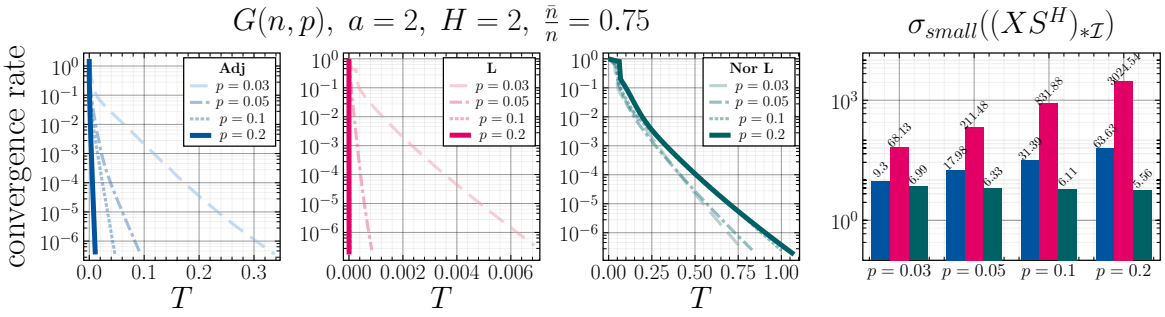


Figure 3: Convergence rate of the relative loss $\frac{\mathcal{L}(\boldsymbol{W}(T))-\tilde{\mathcal{L}}_H}{\mathcal{L}(\boldsymbol{W}(0))-\tilde{\mathcal{L}}_H}$ in the gradient flow training for $G(n, p)$ model, $n = 200$ and varying p. Panes demonstrate loss flows for the different choices of the shift operator S (left to right: adjacency matrix, graph Laplacian, normalized Laplacian) and the singular values $\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right)$; for 4 different values of p.

Note that in terms of the choice of shift operator, the convergence rate in Figure 3 generally supports the ordering established in Figure 2: gradient flows with unnormalized Laplacian tend to converge overall faster than ones using adjacency matrix, whilst the case of normalized Laplacian remains remarkably stagnant. For both unnormalized operators,

convergence rate tends to suffer from sparser graph structures with the effects noticeably worse for the graph Laplacian; on the contrary, in the case of the normalized Laplacian operator, denser graphs may result into the slower convergence rate supported by the smaller values of $\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right)$.

Remark 6.1 (Normalized gradient flow for larger and denser graphs)**.** Linear GNN training in Figure 3 is facilitated through the discrete integration of the gradient flow (3.1). Since such integration is done in the direction of the anti-gradient, one aims to preserve the non-decreasing nature of the loss $\mathcal{L}(\boldsymbol{W}(T))$ along the trajectory by appropriately choosing the integration step which may be forced to be unfeasibly small. Consequently, the explosive growth of the singular value $\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right)$ yields the faster convergence in time T (Theorem 3.1), but, at the same time, it poorly affects the efficiency of the numerical integration due to the blow-up of gradient norms resulting into the smaller stepsizes, especially for larger and denser graphs with unnormalized shift operators.

One possible way to avoid the diminishing stepsizes is to consider instead the normalized gradient flow, $\frac{dW_\ell(t)}{dt} = -\frac{\nabla_{W_\ell}\mathcal{L}\left(\boldsymbol{W}(t)\right)}{\left\|\nabla_{W_\ell}\mathcal{L}\left(\boldsymbol{W}(t)\right)\right\|_F}$, allowing for a higher number of nodes and edges in graph G; we provide example for $G(n,p)$ model in the case of $n = 500$ in Figure 4. Note that whilst the normalized flow is expected to converge to the same global optimum, one loses the guarantee of exponential convergence. Besides that, we posit that qualitative results and aspects of convergence remain the same for both normalized and unnormalized gradient flows, and further provide results only for the unnormalized integration to maintain the theoretical framework of Theorem 3.1.

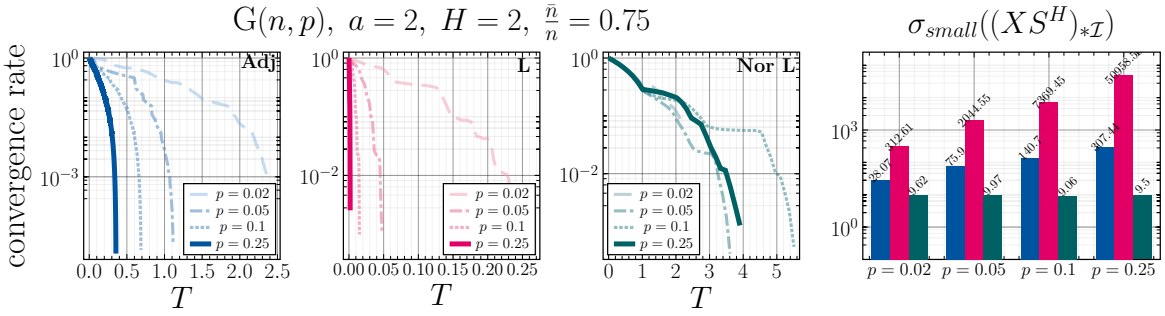


Figure 4: Convergence rate of the relative loss $\frac{\mathcal{L}(\boldsymbol{W}(T))-\tilde{\mathcal{L}}_H}{\mathcal{L}(\boldsymbol{W}(0))-\tilde{\mathcal{L}}_H}$ in the *normalized* gradient flow training for $G(n,p)$ model, $n = 500$ and varying p. Panes demonstrate loss flows for the different choices of the shift operator S (left to right: adjacency matrix, graph Laplacian, normalized Laplacian) and the singular values $\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right)$; for 4 different values of p.

k-nearest neighbors graph, $K_k(n)$. In the k-regular structure of $K_k(n)$, the convergence rate for the adjacency matrix is much closer to the case of the normalized graph Laplacian operator, Figure 5. Moreover, the moderate values of the common degree k result into virtually the same loss trajectory during training, whilst the sparsest case is noticeably slower for unnormalized operators and faster for the normalized Laplacian.

Stochastic Block Model, SBM(n_1, n_2, p, q). In comparison with the simple Erdős–Rényi graph, we consider SBM model with $n_1 : n_2 = 1 : 2$ blocks, a fixed edge sparsity p inside each block and the varying probability q corresponding to the inter-cluster edges (starting from the barely connected clusters up to an almost joined structure resembling $G(n,p)$), Figure 6. Noticeably, the singular value $\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right)$ is sufficiently less affected by the changes in q then by the changes in p in the case of $G(n,p)$, Figure 3, with the adjacency matrix showing

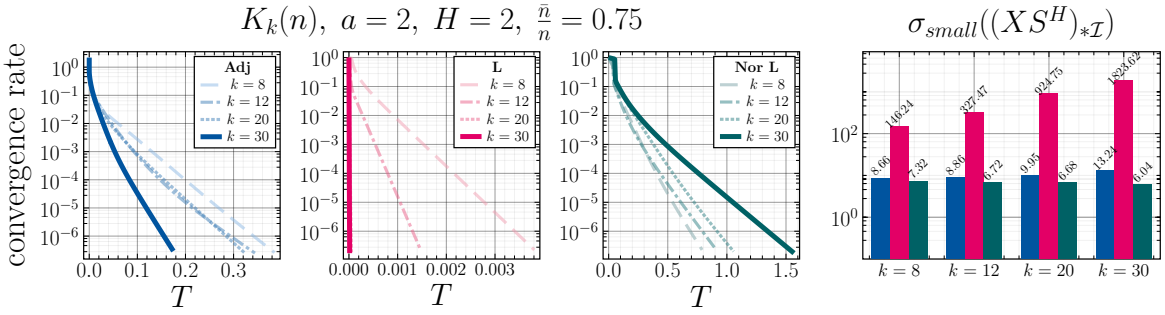


Figure 5: Convergence rate of the relative loss $\frac{\mathcal{L}(\boldsymbol{W}(T))-\tilde{\mathcal{L}}_H}{\mathcal{L}(\boldsymbol{W}(0))-\tilde{\mathcal{L}}_H}$ in the gradient flow training for $K_k(n)$ model, $n=500$ and varying common degree k. Panes demonstrate loss flows for the different choices of the shift operator S(left to right: adjacency matrix, graph Laplacian, normalized Laplacian) and the singular values $\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right)$; for 4 different values of k.

the largest distinction. As a result, the convergence rate for the SBM model is influenced by the inter-block connections during training only for larger q when the graph is well mixed for both adjacency matrix and unnormalized Laplacian.

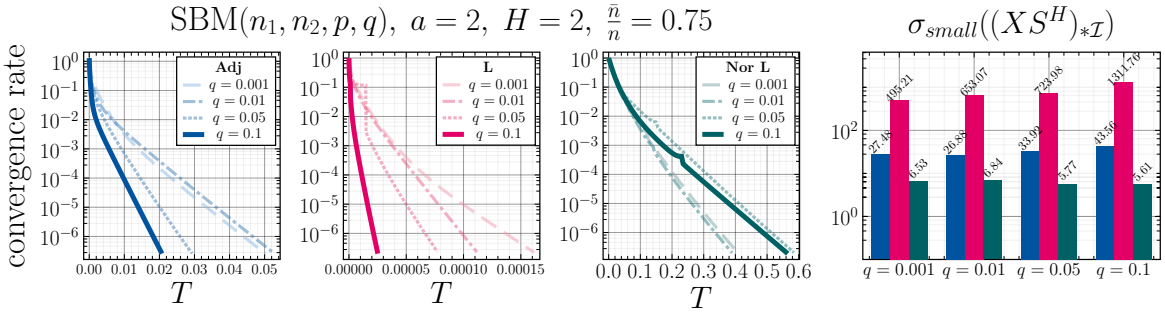


Figure 6: Convergence rate of the relative loss $\frac{\mathcal{L}(\boldsymbol{W}(T))-\tilde{\mathcal{L}}_H}{\mathcal{L}(\boldsymbol{W}(0))-\tilde{\mathcal{L}}_H}$ in the gradient flow training for $\mathrm{SBM}(n_1,n_2,p,q)$ model, $n_1=66$, $n_2=134$, $p=0.15$ and varying q. Panes demonstrate loss flows for the different choices of the shift operator S(left to right: adjacency matrix, graph Laplacian, normalized Laplacian) and the singular values $\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right)$; for 4 different values of q.

Barabási-Albert model, BA(n,m). Finally, scale-free model $\mathrm{BA}(n,m)$ noticeably breaks established pattern: in the case of small minimal degree values m (e.g. in the situations with many hanging nodes), the adjacency matrix exhibits the principle singular value $\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right)$ on par with the smallest one amongst chosen shift operators. Moreover, the overall convergence of the loss maintains the previously established pattern of dependency on the network's density for adajcency matrix and Laplacian operator, whilst the convergence for the normalized Laplacian is virtually independent on the changing network structure.

6.2 Convergence of gradient flow training on real-world graphs

In this subsection, we illustrate the gradient dynamics using a real-world graph dataset. We consider the graph structure with 3107 nodes, corresponding to counties of the United States, and edges representing counties that share borders. The features data X, comprise land surface temperature, precipitation, sunlight, and fine particulate matter, and label data Y, represent air temperature, collected from CDC climate data for each month, averaged over 2008 [1]. The datasets are normalized to have zero mean and unit standard deviation.

We now apply the gradient flow training on the graph neural network associated with the CDC climate data, and endorse Theorem 3.1. In our numerical simulation, we assume that

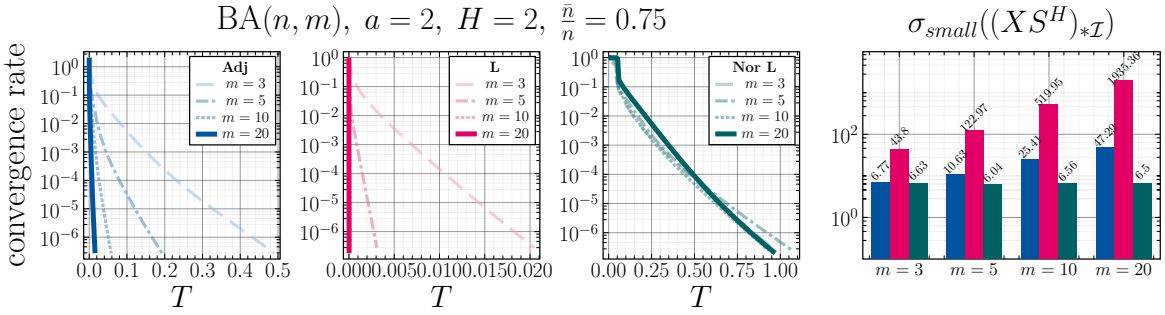
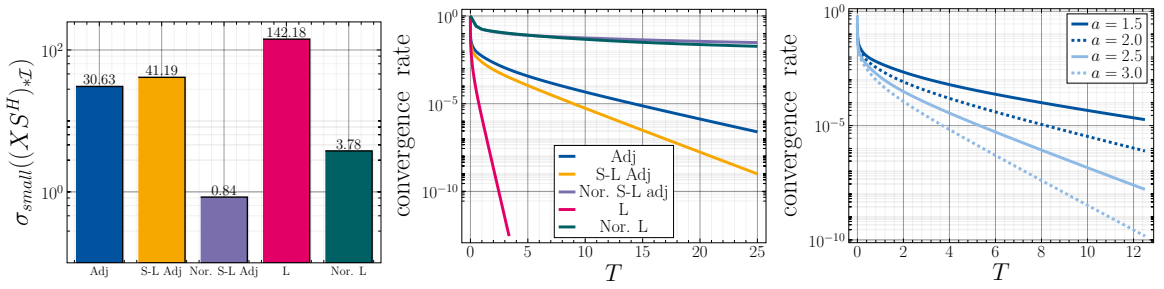


Figure 7: Convergence rate of the relative loss $\frac{\mathcal{L}(\boldsymbol{W}(T))-\tilde{\mathcal{L}}_H}{\mathcal{L}(\boldsymbol{W}(0))-\tilde{\mathcal{L}}_H}$ in the gradient flow training for BA(n, m) model, $n = 200$ and varying minimal degree m. Panes demonstrate loss flows for the different choices of the shift operator S(left to right: adjacency matrix, graph Laplacian, normalized Laplacian) and the singular values $\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right)$; for 4 different values of m.

75% of the node label data are known and $\mathcal{I}$ be the index set of such nodes. We consider a linear graph neural network defined in (2.1) with $H = 2$ and the initial weight matrices: $W_1 \in \mathbb{R}^{32\times 48}$ is a zero matrix, $W_2 \in \mathbb{R}^{32\times 32}$ is the identity matrix, and $W_3 \in \mathbb{R}^{12\times 32}$ has diagonal entries 1.5 and all other 0. Figure 8 shows that the rate of convergence, i.e., the ratio of the relative MSE at time T and the initial relative MSE, is accelerated by the smallest non-zero singular value of $(XS^H)_{*\mathcal{I}}$. In the experiment, we consider different aggregate matrix S of the graph as given in the subsection 6.1.1.

The initial choice of weight matrices proportionate the rate of convergence of MSE to the global minimum. In particular, we consider a 2-layer linear GNN with aggregation matrix S as adjacency matrix of the graph, initial weight matrices $W_1 \in \mathbb{R}^{32\times 48}$ is a zero matrix and W_2 is the identity matrix of order 32. Consequently, Figure 8c illustrates that as the initial choice of diagonal entries of W_3 increases, so does the convergence rate. Although the convergence rate in gradient flow training is proportional to $\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right)$ and smallest singular value of initial weight matrices, graph neural networks are usually trained iteratively in practice. The iteration step must be sufficiently small depending on significant values of $\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right)$ and $\sigma_{\min}(W_i(0))$.



(a) $\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right)$ (b) Convergence rate vs Aggregation matrix (c) Convergence rate vs Initial weight

Figure 8: Comparison of convergence rate in GNNs training for different aggregation matrices. Left: Chart of smallest non-zero singular value of $(XS^H)_{*\mathcal{I}}$ for different S. Middle: In gradient flow training of GNNs with fixed initial weight matrices, convergence rate is proportional to $\sigma_{\text{small}}\left((XS^H)_{*\mathcal{I}}\right)$. Right: Convergence rate of a fixed aggregation matrix depends on initial weight matrices.

7 Conclusion

We study the convergence of gradient dynamics in linear graph neural networks. In order to avoid the existence of any sub-local minimum, we focus our study on GNNs without bottleneck layer. We demonstrate that, with appropriate initialization, the square loss converges to the global minimum at an exponential rate during gradient flow training of linear GNNs. Furthermore, a balanced initialization minimizes the total energy of the weight parameters at the global minimum. The convergence rate depends on the aggregation matrix and the initial weights, as confirmed through numerical experiments on synthetic and real-world datasets.

Acknowledgment

The authors acknowledge funding by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - Project number 442047500 through the Collaborative Research Center "Sparsity and Singular Structures" (SFB 1481).

References

[1] North America Land Data Assimilation System (NLDAS). CDC WONDER Online Database, http://wonder.cdc.gov/NASA-NLDAS.html, 2013. accessed on 10th April 2024.

[2] C. C. Aggarwal. *Neural networks and deep learning*, volume 10. Springer, 2018.

[3] S. Arora, N. Cohen, N. Golowich, and W. Hu. A convergence analysis of gradient descent for deep linear neural networks. In *International Conference on Learning Representations*, 2019.

[4] S. Arora, N. Cohen, and E. Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. In *International Conference on Machine Learning*, pages 244–253. PMLR, 2018.

[5] P. Awasthi, A. Das, and S. Gollapudi. A convergence analysis of gradient descent on graph neural networks. *Advances in Neural Information Processing Systems*, 34:20385–20397, 2021.

[6] B. Bah, H. Rauhut, U. Terstiege, and M. Westdickenberg. Learning deep linear neural networks: Riemannian gradient flows and convergence to global minimizers. *Information and Inference: A Journal of the IMA*, 11(1):307–353, 2022.

[7] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[8] P. Bartlett, D. Helmbold, and P. Long. Gradient descent with identity initialization efficiently learns positive definite linear transformations by deep residual networks. In *International Conference on Machine Learning*, pages 521–530. PMLR, 2018.

[9] S. P. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[10] S. Brin. The PageRank citation ranking: Bringing order to the web. *Proceedings of ASIS*, 98:161–172, 1998.

[11] A. Brutzkus and A. Globerson. Globally optimal gradient descent for a ConvNet with Gaussian inputs. In *International Conference on Machine Learning*, pages 605–614. PMLR, 2017.

[12] S. Chatterjee. Convergence of gradient descent for deep neural networks. *arXiv preprint, arXiv:2203.16462*, 2022.

[13] M. Cheng and H. Xu. A Quasi-Wasserstein loss for learning graph neural networks. *arXiv preprint, arXiv:2310.11762*, 2023.

[14] Y. Chitour, Z. Liao, and R. Couillet. A geometric approach of gradient descent algorithms in linear neural networks. *Mathematical Control and Related Fields*, 13(3):918–945, 2023.

[15] Z. Cui, K. Henrickson, R. Ke, and Y. Wang. Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting. *IEEE Transactions on Intelligent Transportation Systems*, 21(11):4883–4894, 2019.

[16] G. E. Dahl, D. Yu, L. Deng, and A. Acero. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):30–42, 2011.

[17] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in Neural Information Processing Systems*, 29, 2016.

[18] W. Dong, J. Wu, X. Zhang, Z. Bai, P. Wang, and M. Woźniak. Improving performance and efficiency of graph neural networks by injective aggregation. *Knowledge-Based Systems*, 254:109616, 2022.

[19] P. Erdos, A. Rényi, et al. On the evolution of random graphs. *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, 5(1):17–60, 1960.

[20] M. Essid and J. Solomon. Quadratically regularized optimal transport on graphs. *SIAM Journal on Scientific Computing*, 40(4):A1961–A1986, 2018.

[21] E. Facca and M. Benzi. Fast iterative solution of the optimal transport problem on graphs. *SIAM Journal on Scientific Computing*, 43(3):A2295–A2319, 2021.

[22] J. Gasteiger, A. Bojchevski, and S. Günnemann. Predict then propagate: Graph neural networks meet personalized PageRank. In *International Conference on Learning Representations*, 2019.

[23] R. Ge, F. Huang, C. Jin, and Y. Yuan. Escaping from saddle points—online stochastic gradient for tensor decomposition. In *Conference on Learning Theory*, pages 797–842. PMLR, 2015.

[24] R. Ge, J. D. Lee, and T. Ma. Learning one-hidden-layer neural networks with landscape design. In *6th International Conference on Learning Representations*. ICLR 2018, 2018.

[25] W. Hamilton, Z. Ying, and J. Leskovec. Inductive representation learning on large graphs. *Advances in Neural Information Processing Systems*, 30, 2017.

[26] W. L. Hamilton. *Graph representation learning*. Morgan & Claypool Publishers, 2020.

[27] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[28] P. W. Holland, K. B. Laskey, and S. Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109–137, 1983.

[29] C. R. Johnson and R. A. Horn. *Matrix analysis*. Cambridge University Press, 1985.

[30] K. Kawaguchi. Deep learning without poor local minima. *Advances in Neural Information Processing Systems*, 29, 2016.

[31] S. Kearnes, K. McCloskey, M. Berndl, V. Pande, and P. Riley. Molecular graph convolutions: Moving beyond fingerprints. *Journal of Computer-Aided Molecular Design*, 30:595–608, 2016.

[32] T. Laurent and J. Brecht. Deep linear networks with arbitrary loss: All local minima are global. In *International Conference on Machine Learning*, pages 2902–2907. PMLR, 2018.

[33] J. Lee, Y. Oh, Y. In, N. Lee, D. Hyun, and C. Park. GraFN: Semi-supervised node classification on graph with few labels via non-parametric distribution assignment. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2243–2248, 2022.

[34] J. D. Lee, M. Simchowitz, M. I. Jordan, and B. Recht. Gradient descent only converges to minimizers. In *Conference on Learning Theory*, pages 1246–1257. PMLR, 2016.

[35] Y. Li and Y. Yuan. Convergence analysis of two-layer neural networks with ReLU activation. *Advances in Neural Information Processing Systems*, 30, 2017.

[36] K. G. Murty and S. N. Kabadi. Some NP-complete problems in quadratic and nonlinear programming. Technical Report 2, 1987.

[37] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations*, 2017.

[38] Y. Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.

[39] T. Pham and X. Li. Neural network-based power flow model. In *2022 IEEE Green Technologies Conference (GreenTech)*, pages 105–109. IEEE, 2022.

[40] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

[41] A. Saxe, J. McClelland, and S. Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *Proceedings of the International Conference on Learning Represenatations*. ICLR 2014, 2014.

[42] M. T. Schaub, J.-C. Delvenne, M. Rosvall, and R. Lambiotte. The many facets of community detection in complex networks. *Applied Network Science*, 2:1–13, 2017.

[43] A. Srinivasan, S. Muggleton, R. D. King, and M. J. Sternberg. Mutagenesis: ILP experiments in a non-determinate biological domain. In *Proceedings of the 4th International Workshop on Inductive Logic Programming*, volume 237, pages 217–232. Citeseer, 1994.

[44] M. Trager, K. Kohn, and J. Bruna. Pure and spurious critical points: A geometric study of linear networks. In *8th International Conference on Learning Representations, ICLR*, 2020.

[45] O. Wieder, S. Kohlbacher, M. Kuenemann, A. Garon, P. Ducrot, T. Seidel, and T. Langer. A compact review of molecular property prediction with graph neural networks. *Drug Discovery Today: Technologies*, 37:1–12, 2020.

[46] S. Wu, F. Sun, W. Zhang, X. Xie, and B. Cui. Graph neural networks in recommender systems: A survey. *ACM Computing Surveys*, 55(5):1–37, 2022.

[47] K. Xu, M. Zhang, S. Jegelka, and K. Kawaguchi. Optimization of graph neural networks: Implicit acceleration by skip connections and more depth. In *International Conference on Machine Learning*, pages 11592–11602. PMLR, 2021.

[48] X. Zhan. *Matrix theory*, volume 147. American Mathematical Society, 2013.

[49] K. Zhou, X. Huang, Y. Li, D. Zha, R. Chen, and X. Hu. Towards deeper graph neural networks with differentiable group normalization. *Advances in Neural Information Processing Systems*, 33:4917–4928, 2020.