# Graph Coarsening with Message-Passing Guarantees

**Antonin Joly**
IRISA, Rennes, France
antonin.joly@inria.fr

**Nicolas Keriven**
CNRS, IRISA, Rennes, France
nicolas.keriven@cnrs.fr

## Abstract

Graph coarsening reduces the size of a large graph to decrease computational load and memory footprint, while preserving some of its key properties. For instance, training Graph Neural Networks (GNNs) on coarsened graphs leads to drastic savings in time and memory. However, GNNs rely on the Message-Passing (MP) paradigm, and classical spectral preservation guarantees for graph coarsening do not directly lead to theoretical guarantees when performing naive message-passing on the coarsened graph. In this work, we propose a new message-passing operation specific to coarsened graphs, which exhibit theoretical guarantees on the preservation of the propagated signal. We conduct node classification tasks on synthetic and real data and observe improved results compared to performing naive message-passing on the coarsened graph.

## 1 Introduction

In recent years, several applications in data science and machine learning have produced large-scale *graph* data [2, 10]. To handle such massive graphs, researchers have developed general-purpose *graph reduction* methods [1], such as **graph coarsening** [3, 16]. It consists in producing a small graph from a large graph while retaining some of its key properties,and starts to play an increasingly prominent role in machine learning applications [3].

**Graph Neural Networks.** Machine Learning on graphs is now largely done by Graph Neural Networks (GNNs) [2, 14, 19]. GNNs are deep architectures on graph that rely on the **Message-Passing** (MP) paradigm [8]: at each layer, the representation $H_i^l \in \mathbb{R}^{d_l}$ of each node $1 \leq i \leq N$, is updated by *aggregating* and *transforming* the representations of its neighbours at the previous layer $\{H_j^{l-1}\}_{j \in \mathcal{N}(i)}$, where $\mathcal{N}(i)$ is the neighborhood of $i$. In most examples, this aggregation can be represented as a *multiplication* of the node representation matrix $H^{l-1} \in \mathbb{R}^{N \times d_{l-1}}$ by a *propagation matrix* $S \in \mathbb{R}^{N \times N}$ related to the graph structure, followed by a fully connected transformation. That is, starting with initial node features $H^0$, the GNN $\Phi_\theta$ outputs after $k$ layers:

$$H^l = \sigma\left(SH^{l-1}\theta_l\right), \quad \Phi_\theta(H^0, S) = H^k, \tag{1}$$

where $\sigma$ is an activation function applied element-wise (often ReLU), $\theta_l \in \mathbb{R}^{d_{l-1} \times d_l}$ are learned parameters and $\theta = \{\theta_1, \dots, \theta_k\}$. We emphasize here the dependency of the GNN on the propagation matrix $S$. Classical choices include mean aggregation $S = D^{-1}A$ or the normalized adjacency $S = D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$, with $A$ the adjacency matrix of the graph and $D$ the diagonal matrix of degrees. An interesting example is the Simplified Graph Convolution (SGC) model [20], which consists in removing all the non-linearity ($\sigma = id$). SGC reaches quite good performances when compared to GNNs and due to its simplicity, it has been extensively employed in theoretical analyses [13, 22].

In this paper, we consider graph coarsening as a **preprocessing** step to downstream tasks [6, 11], in opposition to e.g. graph *pooling* [21] within a GNN itself, which is often data-driven and fully differentiable [21]. A primary question is the following: after coarsening, **is training a GNN on a coarsened graph provably close to training it on the original graph?**

There are many criteria to measure the quality of graph coarsening algorithms [3, 5, 16]. A classical objective is the preservation of *spectral properties* of the graph Laplacian [1, 4, 12, 16, 17]. Loukas

A. Joly, N. Keriven, Graph Coarsening with Message-Passing Guarantees (Extended Abstract). Presented at the Third Learning on Graphs Conference (LoG 2024), Virtual Event, November 26–29, 2024.

[16] materializes this by the so-called *Restricted Spectral Approximation* (RSA, see Sec. 2) property. Surprisingly, the RSA *does not generally lead to guarantees on the MP process* at the core of GNNs, even for very simple signals. In this paper, we address this problem by defining a **new propagation matrix** $S_c^{\text{MP}}$ *specific to coarsened graphs*, which translate the RSA bound to MP guarantees. The proposed matrix $S_c^{\text{MP}}$ can be computed for any given coarsening and **is not specific to the coarsening algorithm used to produce it** , as long as it produces coarsenings with RSA guarantees. To our knowledge, the only previous work to propose a new propagation matrix for coarsened graphs is [11], where the authors obtain guarantees for a specific GNN model (APPNP [15]), which is quite different from generic MP.

**Notations.** For a matrix $Q \in \mathbb{R}^{n\times N}$, the matrix $Q^+ \in \mathbb{R}^{N\times n}$ is its pseudo-inverse. For a symmetric positive semi-definite (p.s.d.) matrix $L \in \mathbb{R}^{N\times N}$, and $x \in \mathbb{R}^N$ we denote by $\|x\|_L = \sqrt{x^\top L x}$ the Mahalanobis semi-norm associated to $L$. For a matrix $P \in \mathbb{R}^{N\times N}$, we denote by $\|P\| = \max_{\|x\|=1} \|Px\|$ the operator norm of $P$, and $\|P\|_L = \|L^{\frac{1}{2}} P L^{-\frac{1}{2}}\|$. For a subspace $R$, we say that a matrix $P$ is $R$-preserving if $x \in R$ implies $Px \in R$. Finally, for a matrix $X \in \mathbb{R}^{N\times d}$, we denote its columns by $X_{:,i}$, and define $\|X\|_{:,L} = \sum_i \|X_{:,i}\|_L$.

## 2 Background on Graph Coarsening

A graph $G$ with $N$ nodes is described by its weighted adjacency matrix $A \in \mathbb{R}^{N\times N}$. $L \in \mathbb{R}^{N\times N}$ refers to the symmetric p.s.d. Laplacian of the graph, which can either be the combinatorial Laplacian $L = D - A$, or the symmetric normalized Laplacian $L = I_N - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$. We denote by $L \in \mathbb{R}^{N\times N}$ a notion of symmetric p.s.d. Laplacian of the graph: classical choices include the combinatorial Laplacian $L = D - A$ or the symmetric normalized Laplacian $L = I_N - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$.

**Coarsening matrix.** A coarsening algorithm takes a graph $G$ with $N$ nodes, and produces a coarsened graph $G_c$ with $n < N$ nodes. Intuitively, nodes in $G$ are grouped in "super-nodes" in $G_c$, This mapping can be represented *via* a **coarsening matrix** $Q \in \mathbb{R}^{n\times N}$:

$$Q = \begin{cases} Q_{ki} > 0 & \text{if the } i\text{-th node of } G \text{ is mapped to the } k\text{-th super-node of } G_c \\ Q_{ki} = 0 & \text{otherwise} \end{cases}$$

The **lifting matrix** is the pseudo-inverse of the coarsening matrix $Q^+$, and plays the role of inverse mapping from the coarsened graph to the original one. The **coarsening ratio** $r$ is: $r = 1 - \frac{n}{N}$. In this paper, as in the majority of the literature [16], we consider only **well-mapped** coarsenings, where nodes in $G$ are mapped to a unique node in $G_c$ (i.e. $Q$ has exactly one non-zero value per column).

**Restricted Spectral Approximation.** Preserving spectral properties is key in graph coarsening. For example, this is formalized in [16] with the *Restricted Spectral Approximation* (RSA), which measures how much the projection $\Pi = Q^+ Q$ is close to the identity for a class of signals. More precisely, for a signal $x \in \mathbb{R}^N$ over the nodes of $G$, $x_c \in \mathbb{R}^n$ and $\tilde{x} \in \mathbb{R}^N$, where

$$x_c = Qx, \qquad \tilde{x} = Q^+ x_c = \Pi x, \tag{2}$$

are the coarsened and re-lifted signals. The *RSA constant* defined below quantifies the RSA.

**Definition 1 (Restricted Spectral Approximation constant)** *Consider a subspace $\mathcal{R} \subset \mathbb{R}^N$, a Laplacian $L$, a coarsening matrix $Q$ and its corresponding projection operator $\Pi = Q^+ Q$. The* RSA constant $\epsilon_{L,Q,\mathcal{R}}$ *is defined as*

$$\epsilon_{L,Q,\mathcal{R}} = \sup_{x\in\mathcal{R}, \|x\|_L=1} \|x - \Pi x\|_L \tag{3}$$

In other words, the RSA constant measures how much signals in $\mathcal{R}$ are preserved by the coarsening-lifting operation, with respect to the norm $\|\cdot\|_L$. In practice, $\mathcal{R}$ is often chosen a the subspace spanned by the first eigenvectors of $L$ ordered by increasing eigenvalue. Given some $\mathcal{R}$ and Laplacian $L$, the goal of a coarsening algorithm is then to produce a coarsening $Q$ *with the smallest RSA constant possible*. While the "best" coarsening $\arg\min_Q \epsilon_{L,Q,\mathcal{R}}$ is generally computationally unreachable, there are many possible heuristic algorithms, often based on greedy merging of nodes [16].

## 3 Message-Passing on coarsened graphs

We have seen that coarsening algorithms generally aim at preserving the spectral properties of the graph Laplacian, leading to small RSA constants $\epsilon_{L,Q,\mathcal{R}}$. However, this generally does not directly translate to guarantees on the MP process materialized by the matrix $S$. In this section, we propose a new propagation matrix such that **small RSA constants leads to preserved MP**, which then leads to guarantees for training GNNs on coarsened graphs.

**A new propagation matrix on coarsened graphs.** Given a graph $G$ with a propagation matrix $S$ and a coarsened graph $G_c$ with a coarsening matrix $Q$, our goal is to define a propagation matrix $S_c^{\text{MP}} \in \mathbb{R}^{n\times n}$ such that one round of MP on the coarsened signal $x_c$ followed by lifting is close to performing MP in the original graph: $Q^+ S_c^{\text{MP}} x_c \approx Sx$. For instance, assuming that the propagation matrix $S = f_S(A)$ is the output of a function $f_S$ of the graph's adjacency matrix, the most natural choice, often adopted in the literature [6], is to simply take $S_c = f_S(A_c)$, where $A_c$ is the adjacency matrix of the coarsened graph. However, this does not generally leads to the desired guarantees. Some authors propose different variant of $S_c$ adapted to specific cases [11, 21] (see Sec. 4), but none offers generic message-passing guarantees. To address this, we propose a new propagation matrix:

$$S_c^{\text{MP}} = QSQ^+ \in \mathbb{R}^{n\times n}\,. \tag{4}$$

Despite the simplicity of this expression, we will see that under some mild hypotheses this choice indeed leads to preservation guarantees of message-passing for coarsenings with small RSA constants. An important remark is that, unlike all the examples in the literature, the proposed matrix $S_c^{\text{MP}}$ **may be asymmetric even when $S$ is symmetric**, unless $Q$ is orthogonal and $Q^\top = Q^+$, which is not the case for many classical coarsenings [16].

**Message-Passing guarantees.** To state our result, we must make some technical assumptions relating the choice of Laplacian, the nature of the coarsening, and the propagation matrix $S$.

**Assumption 1** *Assume that* $\Pi$ *and* $S$ *are both* $\ker(L)$*-preserving, and that* $S$ *is* $\mathcal{R}$*-preserving.*

**Theorem 1** *Define* $S_c^{\text{MP}}$ *as* (4)*. Under Assumption 1, for all* $x \in \mathcal{R}$*,*

$$\|Sx - Q^+ S_c^{\text{MP}} x_c\|_L \leq \epsilon_{L,Q,\mathcal{R}} \|x\|_L \,(C_S + C_\Pi) \tag{5}$$

*where* $C_S := \|S\|_L$ *and* $C_\Pi := \|\Pi S\|_L$*.*

This theorem shows that the RSA error $\epsilon_{L,Q,\mathcal{R}}$ directly translates to an error bound between $Sx$ and $Q^+ S_c^{\text{MP}} x_c$. To satisfy the assumptions, we choose in our experiment to use GCNconv [14] with $S = D(\hat{A})^{-\frac{1}{2}} \hat{A} D(\hat{A})^{-\frac{1}{2}}$ with $A = A + I_N$, and to compute a coarsening with a good RSA constant for the "Laplacian" $L = (1+\delta)I_N - S$ with small $\delta > 0$ and $\mathcal{R}$ spanned by the first eigenvectors of $L$ (the small $\delta$ is used to conveniently reduce the kernel of $L$ to $\{0\}$). A broader discussion on the multiplicative constant can be found in the full paper.

**GNN training on coarsened graph.** We now instantiate our message-passing guarantees to GNN training on coarsened graph, with SGC as a primary example. To fix ideas, we consider a single large graph $G$, and a node-level task. Given some node features $X \in \mathbb{R}^{N\times d}$, the goal is to minimize a loss function $J : \mathbb{R}^N \to \mathbb{R}_+$ on the output of a GNN $\Phi_\theta(X, S) \in \mathbb{R}^N$ (assumed unidimensional for simplicity) with respect to the parameter $\theta$:

$$\min_{\theta\in\Theta} R(\theta) \text{ with } R(\theta) := J(\Phi_\theta(X, S)) \tag{6}$$

where $\Theta$ is a set of parameters that we assume bounded. Instead, one may want to train on the coarsened graph $G_c$, which can be done by minimizing instead

$$R_c(\theta) := J(Q^+ \Phi_\theta(X_c, S_c^{\text{MP}})) \tag{7}$$

where $X_c = QX$. That is, the GNN is applied on the coarsened graph, and the output is then lifted to compute the loss, which is then back-propagated to compute the gradient of $\theta$. We make the following assumption to state our result.

**Assumption 2** *Assume that there is a constant* $C_J$ *such that* $|J(x) - J(x')| \leq C_J \|x - x'\|_L$*. Moreover, assume that* $\sigma$ *is* $\mathcal{R}$*-preserving,* $\|\sigma(x) - \sigma(x')\|_L \leq C_\sigma \|x - x'\|_L$*,* $\sigma$ *and* $Q^+$ *commute.*

The first part is valid for most loss functions, while the $\mathcal{R}$-preserving part is, for now, only verified for the $id$ activation function and thus the SGC model. Finding non-trivial activation functions that preserve some subspaces, or replacing this assumption with a more flexible one, are important paths for future work.

**Theorem 2** *Under Assumptions 1 and 2: for all node features $X \in \mathbb{R}^{N\times d}$ such that $X_{:,i} \in \mathcal{R}$, denoting by $\theta^\star = \arg\min_{\theta\in\Theta} R(\theta)$ and $\theta_c = \arg\min_{\theta\in\Theta} R_c(\theta)$, we have*

$$R(\theta_c) - R(\theta^\star) \leq C\epsilon_{L,Q,\mathcal{R}}\|X\|_{:,L} \tag{8}$$

*with $C = 2C_J C_\sigma^k C_\Theta (C_S + C_\Pi)\sum_{l=1}^k \bar{C}_\Pi^{k-l} C_S^{l-1}$ where $\bar{C}_\Pi := \|\Pi S \Pi\|_L$ and $C_\Theta$ is a constant that depends on the parameter set $\Theta$.*

## 4 Experiments

**Setup.** We choose the propagation matrix from GCNconv [14], that is, $S = f_S(A) = D(\hat{A})^{-\frac{1}{2}}\hat{A}D(\hat{A})^{-\frac{1}{2}}$ with $\hat{A} = A + I_N$. We report here the result for SGC [20], which satisfies Assumption 2 (experiment with the true GCNconv [14] in the full paper). As detailed in the previous section, we take $L = (1+\delta)I_N - S$ with $\delta = 0.001$ and $\mathcal{R}$ as the $K$ first eigenvectors of $L$ ($K = N/10$ in our experiments), ensuring that assumption 1 is satisfied. We adapt the algorithm from [16] to coarsen the graphs with a good RSA constant $\epsilon_{L,Q,\mathcal{R}}$, more details in the full paper.

On coarsened graphs, we compare five propagation matrices: $S_c^{\text{MP}} = QSQ^+$, our proposed matrix; $S_c = f_S(A_c)$, the naive choice $S_c^{diag} = \hat{D'}^{-1/2}(A_c + C)\hat{D'}^{-1/2}$, proposed in [11]; $S_c^{diff} = QSQ^\top$, roughly inspired by Diffpool [21] and $S_c^{sym} = (Q^+)^\top SQ^+$, the lifting employed to compute the adjacency matrix of the coarsened graph $A_c = (Q^+)^\top AQ^+$.

**Node classification.** We perform node classification experiments on real-world graphs, namely Cora [18], Citeseer [7] and Reddit [9]. For simplicity, we restrict them to their largest connected component[1], since using a connected graph is far more convenient for coarsening algorithms (details in the full paper). Despite the lifting procedure, training on the coarsened graph results is faster than using the entire graph (e.g., by approximately $30\%$ for a coarsening ratio of $r = 0.5$ when parallelized on GPU). Each classification results is averaged on $10$ random training. Results are reported in Table 1. We observe that the proposed propagation matrix $S_c^{\text{MP}}$ yields better results and is more stable, especially for high coarsening ratio. The detailed hyper-parameters for each model and each dataset can be found with additional comments on the results and datasets in the full paper.

| SGC | Cora | | Citeseer | | Reddit | |
|---|---|---|---|---|---|---|
| $r$ | 0.5 | 0.7 | 0.5 | 0.7 | 0.9 | 0.99 |
| $S_c^{sym}$ | 16.1 ± 3.8 | 16.4 ± 4.7 | 18.6 ± 4.6 | 19.8 ± 5.0 | 37.1 ± 6.6 | 3.7 ± 5.5 |
| $S_c^{diff}$ | 21.8 ± 2.2 | 13.6 ± 2.8 | 30.5 ± 0.2 | 23.1 ± 0.0 | 18.3 ±0.0 | 14.9 ± 0.0 |
| $S_c$ | 78.7 ± 0.0 | 74.6 ± 0.1 | 72.8 ± 0.1 | 72.5 ± 0.1 | 87.5 ± 0.1 | 37.3 ± 0.0 |
| $S_c^{diag}$ | 78.7 ± 0.1 | 77.3 ± 0.0 | 73.4 ± 0.1 | 73.1 ± 0.4 | 87.6 ± 0.1 | 37.3 ± 0.0 |
| $S_c^{\text{MP}}$ **(ours)** | **80.3** ± 0.1 | **78.5** ± 0.0 | **74.6** ± 0.1 | **74.2** ± 0.1 | **90.2** ±0.0 | **64.1** ± 0.0 |
| Full Graph | 81.6 ± 0.1 | | 73.6 ± 0.0 | | 94.9 ± 0.0 | |

**Table 1:** Accuracy in $\%$ for node classification and different coarsening ratio and models.

[1] hence the slight difference with other reported results on these datasets

## References


[1] G. Bravo Hermsdorff and L. Gunderson. A unifying framework for spectrum-preserving graph sparsification and coarsening. *Advances in Neural Information Processing Systems*, 32, 2019. 1

[2] M. M. Bronstein, J. Bruna, T. Cohen, and P. Veličković. Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges. *arXiv:2104.13478*, 2021. URL http://arxiv.org/abs/2104.13478. 1

[3] J. Chen, Y. Saad, and Z. Zhang. *Graph coarsening: from scientific computing to machine learning*, volume 79. Springer International Publishing, 2022. ISBN 4032402100282. doi: 10.1007/s40324-021-00282-x. URL https://doi.org/10.1007/s40324-021-00282-x. 1

[4] Y. Chen, R. Yao, Y. Yang, and J. Chen. A gromov-wasserstein geometric view of spectrum-preserving graph coarsening. In *International Conference on Machine Learning*, pages 5257–5281. PMLR, 2023. 1

[5] I. S. Dhillon, Y. Guan, and B. Kulis. Weighted graph cuts without eigenvectors a multilevel approach. *IEEE transactions on pattern analysis and machine intelligence*, 29(11):1944–1957, 2007. 1

[6] C. Dickens, E. Huang, A. Reganti, J. Zhu, K. Subbian, and D. Koutra. Graph coarsening via convolution matching for scalable graph neural network training. In *Companion Proceedings of the ACM on Web Conference 2024*, pages 1502–1510, 2024. 1, 3

[7] C. L. Giles, K. D. Bollacker, and S. Lawrence. Citeseer: An automatic citation indexing system, 1998. URL www.neci.nj.nec.com. 4

[8] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl. Neural Message Passing for Quantum Chemistry. In *International Conference on Machine Learning (ICML)*, pages 1–14, 2017. ISBN 978-1-4577-0079-8. doi: 10.1002/nme.2457. URL http://arxiv.org/abs/1704.01212. 1

[9] W. L. Hamilton, Z. Ying, and J. Leskovec. Inductive Representation Learning on Large Graphs. In *NIPS*, pages 1024–1034, 2017. 4

[10] W. Hu, M. Fey, M. Zitnik, Y. Dong, H. Ren, B. Liu, M. Catasta, and J. Leskovec. Open Graph Benchmark: Datasets for Machine Learning on Graphs. *Neural Information Processing Systems (NeurIPS)*, (NeurIPS):1–34, 2020. URL http://arxiv.org/abs/2005.00687. 1

[11] Z. Huang, S. Zhang, C. Xi, T. Liu, and M. Zhou. *Scaling up Graph Neural Networks Via Graph Coarsening*, volume 1. Association for Computing Machinery, 2021. ISBN 9781450383325. doi: 10.1145/3447548.3467256. 1, 2, 3, 4

[12] Y. Jin, A. Loukas, and J. JaJa. Graph coarsening with preserved spectral properties. In *International Conference on Artificial Intelligence and Statistics*, pages 4452–4462. PMLR, 2020. 1

[13] N. Keriven. Not too little, not too much: a theoretical analysis of graph (over)smoothing. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. URL http://arxiv.org/abs/2205.12156. 1

[14] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2016. 1, 3, 4

[15] J. Klicpera, A. Bojchevski, and S. Günnemann. Predict then propagate: Graph neural networks meet personalized PageRank. *7th International Conference on Learning Representations, ICLR 2019*, pages 1–15, 2019. 2

[16] A. Loukas. Graph reduction with spectral and cut guarantees. *Journal of Machine Learning Research*, 20(116):1–42, 2019. 1, 2, 3, 4

[17] A. Loukas and P. Vandergheynst. Spectrally approximating large graphs with smaller graphs. In *International conference on machine learning*, pages 3237–3246. PMLR, 2018. 1

[18] A. K. Mccallum, K. Nigam, J. Rennie, and K. Seymore. Automating the construction of internet portals with machine learning, 2000. URL www.campsearch.com. 4

[19] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008. 1

[20] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger. Simplifying graph convolutional networks. In *International conference on machine learning*, pages 6861–6871. PMLR, 2019. 1, 4

[21] Z. Ying. Jiaxuan you, christopher morris, xiang ren, will hamilton, and jure leskovec. hierarchical graph representation learning with differentiable pooling. *Advances in neural information processing systems*, 31:4800–4810, 2018. 1, 3, 4

[22] J. Zhu, R. A. Rossi, A. Rao, T. Mai, N. Lipka, N. K. Ahmed, and D. Koutra. Graph Neural Networks with Heterophily. *35th AAAI Conference on Artificial Intelligence, AAAI 2021*, 12B: 11168–11176, 2021. 1