Improving GNN-Based Multiple Scattering Simulation with Rewiring

Rémi Marsal
U2IS, ENSTA
Institut Polytechnique de Paris
remi.marsal@ensta.fr

Stéphanie Chaillat
Laboratoire POEMS, CNRS, INRIA, ENSTA
Institut Polytechnique de Paris
stephanie.chaillat@ensta.fr

Abstract

The recently proposed PIBNet method introduces a learning-based framework for simulating multiple scattering phenomena. It combines the efficiency of boundary integral representations, which transform three-dimensional problems into two-dimensional ones, with the computational advantages of graph neural networks (GNNs). Thus, PIBNet trains a GNN to estimate the trace of the multiple scattering problem solution, i.e., the solution on the obstacle boundaries. However, since long-range interactions are represented by edges, and only a subset can be included due to computational constraints, GNNs may fail to capture omitted interactions. In this paper, we present a new architecture which leverages a rewiring strategy to address this issue. This increases the number of long-range interactions that are explicitly modeled in a single forward pass. As the number of edges processed at each GNN step remains the same, the computational cost is practically not impacted. We demonstrate improved accuracy through extensive experiments on the PIBNet benchmark.

1 Introduction and Related Works

Multiple scattering problems [Martin, 2006] have been repeatedly addressed using learning-based methods, inspired by the success of physics-informed neural networks [Raissi et al., 2019]. Thus, Hao et al. [2021] propose a conditional neural field in which the conditioning mechanism accounts for the location of a monopole source and Nair et al. [2025] introduce a neural field for a specific multiple scattering problem. These approaches are however limited because they are designed to handle one geometry and tackle only two-dimensional problems.

Building on the ideas of the Boundary Element Method (BEM) Bonnet [1999], PIBNet [**?**] introduces a graph neural network (GNN) approach for simulating multiple scattering phenomena. The BEM is a numerical technique that offers significant computational efficiency, for large-scale problems in unbounded domains, compared to the finite element method (FEM) by reducing the dimensionality of the problem. It is based on a reformulation of the partial differential equation (PDE) as a boundary integral equation (BIE) wherein the unknowns are restricted to the boundary of the problem. The BIE solution is then used in the boundary integral representation to evaluate the PDE solution at any location in the volumetric domain. PIBNet proposes a learning-based method to estimate the solution of a field on the obstacle boundaries as this step is the most computationally demanding part of the BEM. To this end, PIBNet uses graphs to represent obstacle boundaries and to model long-range interactions. While all nodes interact with each other, only a fraction of them can be connected to have a computationally tractable method. To address this issue, PIBNet proposes a hierarchical GNN architecture and leverages a physics-informed edge selection strategy that favors connections between nodes that are likely to interact strongly. Since edge selection is random, the GNN must infer long-range interactions that are not explicitly encoded by edges. Due to the fixed number of

1st Workshop on Differentiable Systems and Scientific Machine Learning @ EurIPS 2025

message-passing steps, over-squashing [Alon and Yahav, 2020] may occur, which can lead to the inaccurate modeling of some long-range interactions.

In this work, we introduce a new GNN method based on PIBNet for simulating multiple scattering problems. Specifically, we propose the use of rewiring, a technique that modifies the topology of a graph [Attali et al., 2024]. We apply rewiring during a forward pass to explicitly model a wider range of long-range interactions. In practice, we sample several graphs to connect distant nodes using the PIBNet edge selection strategy. These graphs are then successively used by the GNN. Since the number of edges processed by the message-passing layers remains unchanged, the computational overhead associated with initializing new edges is minimal. The results indicate that this approach yields improved performance on the PIBNet benchmark.

2 Method

2.1 Preliminaries on PIBNet

In this section, we briefly introduce the PIBNet architecture [**?**], a learning-based method for simulating scattering problems in homogeneous domains with multiple disjoint obstacles.

PIBNet is based on MeshGraphNet [Pfaff et al., 2020], a GNN consisting of multiple processor blocks that successively update the edge and node features of a graph. PIBNet extends this architecture with L resolution levels and feature size expansion in a U-Net style. To this end, PIBNet defines multiple graphs with shared nodes that correspond to all or part of the vertices of the obstacle meshes. The graph edges, however, depend on the graph's role in the PIBNet architecture (e.g., modeling local or long-range interactions, downsampling, or upsampling).

First, given $\Omega = \cup_{i=1}^{n} \Omega_i$, the union of n open bounded sets representing non-overlapping obstacles, the *boundary graph* $\mathcal{G}^0 = (V^0, E^0)$ with nodes V^0 and bidirected edges E^0 is defined from a discretization of Ω. It is used by PIBNet to represent the obstacle boundaries. The PIBNet L-level multiscale architecture then relies on downsampling graphs $\mathcal{G}^{j-1\to j}$ and upsampling graphs $\mathcal{G}^{j\to j-1}$, $0 < j < L$ to transition from one level to another. In downsampling graphs $\mathcal{G}^{j-1\to j}$, all nodes V^{j-1} at level $j-1$ are connected to an evenly distributed node subset $V^j \subset V^{j-1}$, gradually decreasing node resolution. In contrast, the edges point in the opposite direction in upsampling graphs $\mathcal{G}^{j\to j-1}$. At the highest level $L-1$, PIBNet uses a *distant node graph* $\mathcal{G}^{L-1}$ to represent the long-range interactions intra obstacles and inter obstacles. Even at level $L-1$, dealing with dense graphs is untractable in case of large objects or if they are too many. Consequently, in $\mathcal{G}^{L-1}$, each node in V^{L-1} is only connected to a fraction α of the other nodes. Consistent with the physics principle which state that close objects interact more, PIBNet employs a physics-inspired edge selection strategy. Two candidate edges are randomly proposed for each required edge, and the shorter one is selected. During a forward pass, PIBNet first applies N_b processor blocks to graph $\mathcal{G}^0$, then a processor block to each downsampling graph, N_d processor blocks to $\mathcal{G}^{L-1}$, a processor block to each upsampling graph and finally N_b processor blocks to $\mathcal{G}^0$.

2.2 Rewiring mechanism

PIBNet outperforms other learning-based methods developed for solving PDEs or processing point clouds. However, it faces challenges when simulating the long-range interactions at the highest level $L-1$ because it relies on a graph with randomly sampled edges that represent only a fraction of these interactions. Consequently, the GNN may yield inaccurate estimations of the missing long-range interactions due to over-squashing [Alon and Yahav, 2020]. This occurs because the number of processor blocks N_d is fixed and their capacity is limited.

To address this limitation, we propose a new architecture illustrated in fig. 1 which implements a rewiring technique [Attali et al., 2024] to increase the number of explicitly modeled long-range interactions. Instead of using the same distant node graph $\mathcal{G}^{L-1}$ at the highest scale level of PIBNet architecture, we propose to create N_G distant node graphs $\mathcal{G}_1^{L-1}, \ldots, \mathcal{G}_{N_G}^{L-1}$ to which the N_d processor blocks are applied. In practice, this involves sampling new edges to connect distant nodes V^{L-1} at the highest level $L-1$, while keeping the node set unchanged. As the number of edges processed by each processor block remains the same, the impact on the overall computation cost is minimal. The only overhead stems from initializing an increased number of edge features.

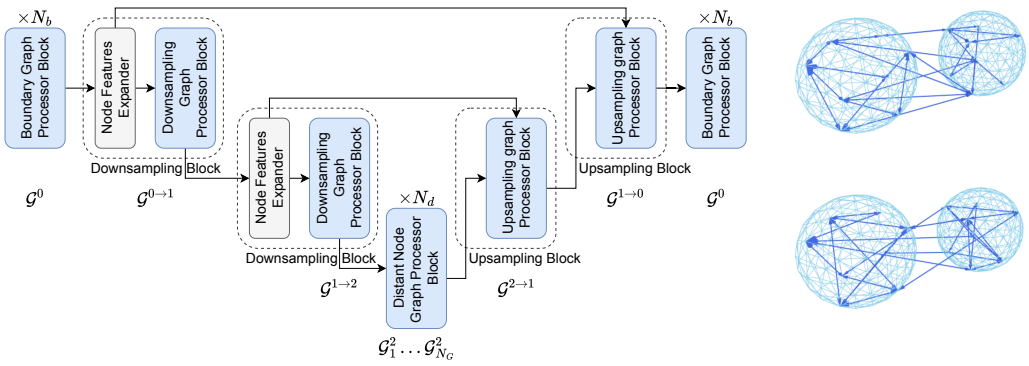

Figure 1: (Left) Illustration of our GNN architecture. The graph representations used by each processor block are shown below the corresponding block. The architecture consists of N_b Boundary Graph Processor Blocks at the beginning, then $L-1$ Downsampling Blocks, N_d Distant Node Graph Processor Blocks, $L-1$ Upsampling Blocks, and N_b Boundary Graph Processor Blocks at the end. (Right) Illustration of two different distant node graphs $\mathcal{G}_i^2$, $\mathcal{G}_j^2$ in dark blue, where $1 \leq i \neq j \leq N_G$. The light blue meshes correspond to the boundary graph $\mathcal{G}^0$ of two obstacles.

3 Experiments

We evaluate our method on **?**'s benchmark, which proposes three multiple scattering problems that can be solved with the BEM. It includes a Laplace problem and two Helmholtz problems, one with Dirichlet boundary conditions and a monopole source and another one with Neumann boundary conditions and an incident plane wave. For each of these problems, a training set and a test set are provided with three obstacles while for the Laplace and the Helmholtz-Dirichlet problem only, additional test sets with six and nine obstacles are available. Performance is evaluated on the predictions of the trace of the field by measuring the relative error $\mathrm{Err}_{\mathrm{rel}}$ for the Laplace problem and the relative error on amplitude $\mathrm{Err}_{\mathrm{ampl}}$ and the absolute error on angle $\mathrm{Err}_{\mathrm{angle}}$ for the Helmholtz problems (for more details, see **?**). For experiments with our method, we adopt the implementation details of **?** and set the number N_G of distant node graphs to three unless otherwise mentioned.

3.1 Results

Table 1: Benchmark results on three obstacle datasets.

Method	Number of Parameters (M)	Laplace	Helmholtz-Dirichlet		Helmholtz-Neumann	
		$\mathrm{Err}_{\mathrm{rel}}$	$\mathrm{Err}_{\mathrm{ampl}}$	$\mathrm{Err}_{\mathrm{angle}}$	$\mathrm{Err}_{\mathrm{ampl}}$	$\mathrm{Err}_{\mathrm{angle}}$
MeshGraphNet	2.4	0.197	0.286	0.191	0.071	0.072
PTv3	77.2	0.066	0.148	0.141	0.065	0.068
Transolver	4.6	0.183	0.596	0.454	0.070	0.073
Transolver++	4.2	0.195	0.635	0.466	0.103	0.122
Erwin	7.9	0.119	0.218	0.175	0.074	0.076
PIBNet	5.2	0.041	0.091	0.090	0.046	0.047
Ours	5.2	**0.036**	**0.084**	**0.083**	**0.045**	**0.045**

We compare our method to the following: MeshGraphNet [Pfaff et al., 2020], Point Transformer v3 [Wu et al., 2024b], Transolver [Wu et al., 2024a], Transolver++ [Luo et al., 2025], Erwin [Zhdanov et al., 2025] and PIBNet [**?**]. The results on the three obstacle datasets, presented in table 1, highlight that our method outperforms all the other approaches, achieving an average improvement of 6.8% relative to PIBNet across all metrics. The model exhibits improved generalization capabilities with six and nine obstacles (see table 2), resulting in better overall performance (an average improvement of 3.4% relative to PIBNet). However, performance varies across problems. Improvements are more pronounced for the Helmholtz-Dirichlet problems compared to the Helmholtz-Neumann problems or to the Laplace problems with six or nine obstacles.

Table 2: Benchmark results on six and nine obstacle datasets.

Method	Laplace		Helmholtz-Dirichlet			
	$\mathrm{Err}_{\mathrm{rel}}$		$\mathrm{Err}_{\mathrm{ampl}}$		$\mathrm{Err}_{\mathrm{angle}}$	
	6 obstacles	9 obstacles	6 obstacles	9 obstacles	6 obstacles	9 obstacles
MeshGraphNet	0.338	0.504	0.552	0.806	0.552	0.806
PTv3	0.254	0.315	0.868	1.066	0.868	1.066
Transolver	0.265	0.344	0.867	1.012	0.867	1.012
Transolver++	0.226	0.383	0.255	0.351	0.255	0.351
Erwin	0.240	0.361	0.417	0.586	0.417	0.586
PIBNet	**0.111**	0.188	0.198	0.300	0.198	0.302
Ours	**0.111**	**0.182**	**0.190**	**0.290**	**0.187**	**0.286**

3.2 Ablation study

Table 3: Performance on the Helmholtz-Dirichlet problem as a function of the number of distant node graphs sampled N_G. When $1 < N_G < N_d$, each graph is processed the same number of times by the processor blocks.

N_G	3 obstacles		6 obstacles		9 obstacles	
	$\mathrm{Err}_{\mathrm{ampl}}$	$\mathrm{Err}_{\mathrm{angle}}$	$\mathrm{Err}_{\mathrm{ampl}}$	$\mathrm{Err}_{\mathrm{angle}}$	$\mathrm{Err}_{\mathrm{ampl}}$	$\mathrm{Err}_{\mathrm{angle}}$
1 (PIBNet)	0.091	0.090	0.198	0.197	0.300	0.302
2	**0.084**	0.083	0.194	0.192	0.300	0.302
3 (Ours)	**0.084**	0.083	**0.190**	0.187	**0.290**	**0.286**
6	**0.084**	**0.082**	0.191	**0.185**	0.292	**0.286**

In table 3, we study the effect of gradually increasing the number of distant node graphs N_G that are used to model long-range interactions, on the Helmholtz Dirichlet problem. We only conduct experiments with N_G values that are factors of the number of processors N_d, such that each graph is used the same number of times. In our case, with $N_d = 6$, our results highlight that the best performance is achieved with $N_G = 3$ and $N_G = 6$. We set $N_G = 3$ in the main experiments for efficiency, as it requires fewer sampled and initialized edges than $N_G = 6$.

Table 4: Comparison to PIBNet with and without ensembling on the Helmholtz-Dirichlet problem. When ensembling is applied, the number of averaged predictions corresponds to the number $N_G = 3$ of sampled distant node graphs by our method to model long-range interactions.

Method	Ensembling	$\mathrm{Err}_{\mathrm{ampl}}$	$\mathrm{Err}_{\mathrm{angle}}$
PIBNet	✗	0.091	0.090
PIBNet	✓	0.085	0.084
Ours	✗	0.084	0.083
Ours	✓	0.079	0.078

Another strategy for considering an increased number of long-range interactions relative to PIBNet is ensembling. However, it is more computationally costly. Thus, in table 4, we compare our approach to ensembling applied on PIBNet where the number of averaged inferences equals the number of distant node graphs ($N_G = 3$), so both approaches deal with the same topologies of long-range interaction modeling. Our method achieves slightly better results than PIBNet with ensembling while requiring almost three times fewer computational resources, as it relies on a single inference. We also show that applying ensembling to our approach improves performance.

4 Conclusion

In this work, we introduce a new learning-based method for solving multiple scattering problems with graph neural networks. Specifically, we upgrade the PIBNet [**?**] architecture by employing a rewiring technique during the forward pass to model a wider range of interactions between distant nodes. Our extensive experiments on the PIBNet benchmark highlight that our approach outperforms PIBNet and other competitive baselines while maintaining the same number of parameters and minimal computational overhead.

5 Acknowledgments

This work is supported by *Agence de l'Innovation de Défense* – AID – via *Centre Interdisciplinaire d'Études pour la Défense et la Sécurité* – CIEDS – (project – APRO)

References

Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. *arXiv preprint arXiv:2006.05205*, 2020.

Hugo Attali, Davide Buscaldi, and Nathalie Pernelle. Rewiring techniques to mitigate oversquashing and oversmoothing in gnns: A survey. *arXiv preprint arXiv:2411.17429*, 2024.

Marc Bonnet. *Boundary Integral Equation Methods for Solids and Fluids*. John Wiley & Sons, 1999.

Wenqu Hao, Yongpin P Chen, Pei-Yao Chen, Ming Jiang, Sheng Sun, and Jun Hu. Solving two-dimensional scattering from multiple dielectric cylinders by artificial neural network accelerated numerical green's function. *IEEE Antennas and Wireless Propagation Letters*, 20(5):783–787, 2021.

Huakun Luo, Haixu Wu, Hang Zhou, Lanxiang Xing, Yichen Di, Jianmin Wang, and Mingsheng Long. Transolver++: An accurate neural solver for pdes on million-scale geometries. *ICML*, 2025.

P. A. Martin. *Multiple Scattering: Interaction of Time-Harmonic Waves with N Obstacles*, volume 107 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 2006.

Siddharth Nair, Timothy F Walsh, Greg Pickrell, and Fabio Semperlotti. Multiple scattering simulation via physics-informed neural networks. *Engineering with Computers*, 41(1):31–50, 2025.

Tobias Pfaff, Meire Fortunato, Alvaro Sanchez-Gonzalez, and Peter Battaglia. Learning mesh-based simulation with graph networks. In *International conference on learning representations*, 2020.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 2019.

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A fast transformer solver for pdes on general geometries. *ICML*, 2024a.

Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer v3: Simpler faster stronger. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4840–4851, 2024b.

Maksim Zhdanov, Max Welling, and Jan-Willem van de Meent. Erwin: A tree-based hierarchical transformer for large-scale physical systems. *ICML*, 2025.