MOX-Report No. 43/2024

Polytopal mesh agglomeration via geometrical deep learning for three-dimensional heterogeneous domains

Antonietti, P.F.; Corti, M., Martinelli, G.

MOX, Dipartimento di Matematica
Politecnico di Milano, Via Bonardi 9 - 20133 Milano (Italy)

mox-dmat@polimi.it https://mox.polimi.it

Polytopal mesh agglomeration via geometrical deep learning for three-dimensional heterogeneous domains*

Paola F. Antonietti[†1], Mattia Corti[‡1], and Gabriele Martinelli[§]

[1]MOX-Dipartimento di Matematica, Politecnico di Milano, Piazza Leonardo da Vinci 32, Milan, 20133, Italy

June 15, 2024

Abstract

Agglomeration techniques are important to reduce the computational costs of numerical simulations and stand at the basis of multilevel algebraic solvers. To automatically perform the agglomeration of polyhedral grids, we propose a novel Geometrical Deep Learning-based algorithm that can exploit the geometrical and physical information of the underlying computational domain to construct the agglomerated grid and simultaneously guarantee the agglomerated grid's quality. In particular, we propose a bisection model based on Graph Neural Networks (GNNs) to partition a suitable connectivity graph of computational three-dimensional meshes. The new approach has a high online inference speed and can simultaneously process the graph structure of the mesh, the geometrical information of the mesh (e.g. elements' volumes, centers' coordinates), and the physical information of the domain (e.g. physical parameters). Taking advantage of this new approach, our algorithm can agglomerate meshes of a domain composed of heterogeneous media in an automatic way. The proposed GNN techniques are compared with the k-means algorithm and METIS: standard approaches for graph partitioning that are meant to process only the connectivity information on the mesh. We demonstrate that the performance of our algorithms outperforms available approaches in terms of quality metrics and runtimes. Moreover, we demonstrate that our algorithm also shows a good level of generalization when applied to more complex geometries, such as three-dimensional geometries reconstructed from medical images. Finally, the capabilities of the model in performing agglomeration of heterogeneous domains are tested in the framework of problems containing microstructures and on a complex geometry such as the human brain.

1 Introduction

Nowadays, the numerical solution of differential problems is central to all fields of engineering and applied sciences, moreover, it is expanding its boundaries to other fields which can benefit from numerical simulations. The applications in all these fields often involve complex physical domains, possibly featuring moving geometries and interfaces, heterogeneous media, immersed interfaces [1], and complex structures [2, 3]. The need to deal with such complex structures leads to the development of new and more sophisticated Finite Element Methods (FEM) that can employ general polygons and polyhedra as grid elements for the numerical discretization of Partial Differential Equations (PDEs). We mention the Polyhedral Discontinuous Galerkin method [4, 5, 6, 7, 8, 9], the Virtual Element Method [10, 11, 12, 13, 14], the Hybrid High-Order method [15, 16, 17, 18], the mimetic finite difference method [19, 20, 21, 22], and the hybridizable discontinuous Galerkin method [23, 24, 25, 26].

The flourishing of these methods takes with it the possibility of an intensive usage of mesh agglomeration techniques, that allow to merge mesh elements to obtain coarser polytopal grids. However, the issue of efficiently performing agglomeration of polytopal meshes is still an open problem and object of intense research [27, 28].

*__Funding__: This research has been funded by the European Union (ERC, NEMESIS, project number 101115663). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. PFA has been partially funded by PRIN2020 n. 20204LN5N5 "Advanced polyhedral discretisations of heterogeneous PDEs for multiphysics problems" research grant, funded by the Italian Ministry of Universities and Research (MUR). The present research is part of the activities of "Dipartimento di Eccelllenza 2023-2027". PFA and MC are members of INdAM-GNCS.

[†]paola.antonietti@polimi.it

[‡]mattia.corti@polimi.it

[§]gabriele2.martinelli@mail.polimi.it

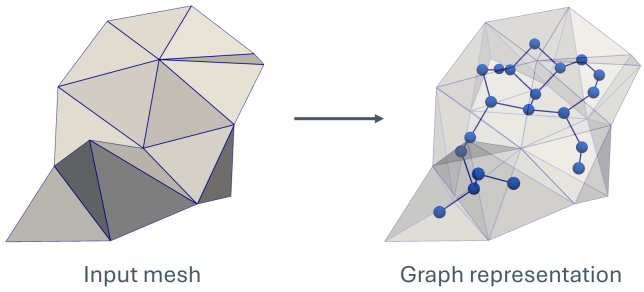


Figure 1: Graph extraction process: input mesh (left) and associated graph representation (right).

Mesh agglomeration is a powerful tool because it can reduce the mesh computational complexity while retaining a detailed geometrical description of the computational domain and/or interfaces. In scientific simulations, reducing the number of elements of a computational mesh can lead to faster simulations and quicker results while maintaining accuracy for example locally increasing the polynomial order. Furthermore, agglomerated meshes stand at the basis of generating a hierarchy of (nested) coarser grids starting from a fine mesh of a complex physical domain of interest. Then such grids' sequence, characterized by different refinement levels, can be employed in multi-grid solvers [29, 30, 31, 32, 33, 34, 35] to accelerate the numerical solution of the underlying algebraic system.

Because of the impossibility of the agglomerated grids to be used in the framework of classical FEMs, grid agglomeration is a quite unexplored topic. Currently employed agglomeration methods rely on iterative algorithms based on graph partitioning of the connectivity matrix, which make them not particularly feasible for complicated or heterogeneous domains or whenever the underlying PDE models multiple physical processes, that have to be properly taken into account in the agglomeration process. Moreover, during the agglomeration procedure, the quality preservation of the underlying mesh is fundamental because it might affect the overall stability and accuracy of the numeric method. Indeed, a suitably agglomerated mesh can give the same accuracy of the solution computed on the finer grid, but with less degrees of freedom. This leads to memory savings and lower computational loads.

In recent years the use of Machine Learning (ML) techniques to accelerate numerical methods has become widespread [36, 37, 38, 39, 40, 41, 42, 43, 44, 45]. In this work, we present a novel geometric deep-learning approach to address the problem of agglomerating three-dimensional meshes of heterogeneous domains. The proposed approach is based on Graph Neural Networks (GNNs). Indeed, we do not simply try to decide an *a-priori* criterium to agglomerate the mesh, which would inevitably result in poor performance or high computational cost due to the impossibility of capturing all the possible configurations. Instead, our ML strategy exploits and processes automatically the huge amount of available data to learn only the distribution of the features of interest for the application, leading to high performance and computational efficiency.

In our approach, the problem of mesh agglomeration is re-framed as a graph partitioning problem by exploiting the structure of the mesh. More precisely, the graph representation of the mesh is obtained by assigning a node to each mesh element. Edges connect pairs of nodes relative to adjacent mesh elements (see Figure 1). By exploiting such an equivalent representation, we can apply the proposed ML algorithm to solve a node classification problem, where each element is assigned to a cluster, corresponding to an element of the agglomerated mesh.

Furthermore, a powerful ability of the proposed GNNs-based algorithm is the possibility to consider further physical properties of the computational domain during mesh agglomeration. Indeed, by adding new input features, and slightly modifying the network structure (particularly, changing the loss function), we can construct an algorithm that takes into account the heterogeneities of the physical domain or possible physical features (such as e.g. embedded microstructures). By exploiting this idea, we can develop a model that automatically agglomerates heterogeneous domains, without the need to separately agglomerate regions with different material properties, with the need for a post-process to glue the obtained meshes to create the final one. Moreover, this new automatic algorithm can have an even bigger impact if the computational domain exhibits, for example, micro-structures and thus cannot be divided into simpler sub-regions.

In this work, we demonstrate that the proposed GNNs-based agglomeration algorithm has high online inference speed and it can process naturally and simultaneously both the graph structure of mesh and the geometrical information that can be attached to the nodes, such as the elements areas/volumes or their centers' coordinates. The technique is then compared with k-means [46] algorithm, with another ML algorithm that can process only

Algorithm 1 General mesh agglomeration strategy AGGLOMERATE($\mathscr{T}_h, h^*$)

Require: Mesh $\mathscr{T}_h$, target mesh size h^*, and bisection model $\mathscr{M}$.
Ensure: Agglomerated mesh $\mathscr{T}_{h^*}$.
  if diam($\mathscr{T}_h$) $\leq h^*$ **then return** $\mathscr{T}_h$
  else
    Extract the connectivity graph $G = (V, E)$ and features X from $\mathscr{T}_h$.
    $Y \leftarrow \mathscr{M}(G, X)$
    Adjust partition Y.
    Partition $\mathscr{T}_h$ into sub-meshes $\mathscr{T}_h^{(1)}, \mathscr{T}_h^{(2)}$ according to Y.
    $\mathscr{T}_{h^*}^{(1)} \leftarrow$ AGGLOMERATE($\mathscr{T}_h^{(1)}, h^*$)
    $\mathscr{T}_{h^*}^{(2)} \leftarrow$ AGGLOMERATE($\mathscr{T}_h^{(2)}, h^*$)
    $\mathscr{T}_{h^*} \leftarrow$ merge $\mathscr{T}_{h^*}^{(1)}$, $\mathscr{T}_{h^*}^{(2)}$
  end if

the geometrical information, and with METIS [47], a standard approach for graph partitioning that is meant to account only for the graph information on the mesh. The study extends the results of [27], which were not directly extended for agglomerating three-dimensional meshes or domains with complicated structures.

Finally, we remark that the proposed algorithm is general and the training can be done once and for all. More precisely, the models do not depend either on the PDE under investigation nor on the numerical method adopted for the discretization.

The rest of the paper is organized as follows. In Section 2, we present the GNN-agglomeration focusing on homogeneous domains in Section 2.1, and on heterogeneous ones in Section 2.2. In Section 3, we present the numerical results focusing on homogeneous domains, first by measuring the effectiveness of the proposed strategies in terms of computational cost, and quality metrics, then evaluating the generalization capabilities over a complex domain such as the human brain ventricles. In Section 4, we focus on the numerical results of heterogeneous domains, by testing the quality of simple datasets and complex heterogeneous domains: one domain containing microstructures and a human brain. Finally, in Section 5 we draw some conclusions.

2 Mesh agglomeration algorithms via GNN

In this section, we introduce our agglomeration algorithm for three-dimensional meshes, discussing also the treatment of heterogeneous media. Most of agglomeration algorithms tackle the problem by re-framing it into a graph partitioning problem by exploiting the connectivity structure of the mesh. In this work, we will focus on constructing an agglomeration algorithm, based on recursive bisection of the associated graph. This strategy recursively bisects the input mesh's connectivity graph until the agglomerated elements have the desired size, as explained in Algorithm 1.

In the particular case of our strategy, we propose to employ GNN-based bisection models $\mathscr{M}$ that take as input a graph G together with a set of features X attached to each node (e.g. barycentric coordinates, area/volume and eventually physical parameters) and output the vector of probabilities Y of each node to belong to cluster 1 or 2. This strategy has key advantages with respect to the state-of-the-art methods (METIS [47] and k-means [46]), more precisely:

- It can process information about the connectivity graph of mesh elements, geometrical features of the elements, and physical properties of the modeled material. On the contrary, METIS is meant to process only graph-based information, and k-means only geometrical information.
- The online inference on new instances is much faster than METIS and k-means. Indeed, in our case, the main computational burden is faced during the training phase, while the other algorithms perform iterations online to provide an estimate of the solution. Moreover, the training phase is independent of the geometry and of the underlying PDE model, so the training phase is performed once and for all.
- The GNN automatically processes additional information without requiring the user to know how that information can benefit performance. On the contrary, for METIS and k-means this has to be explicitly modeled.

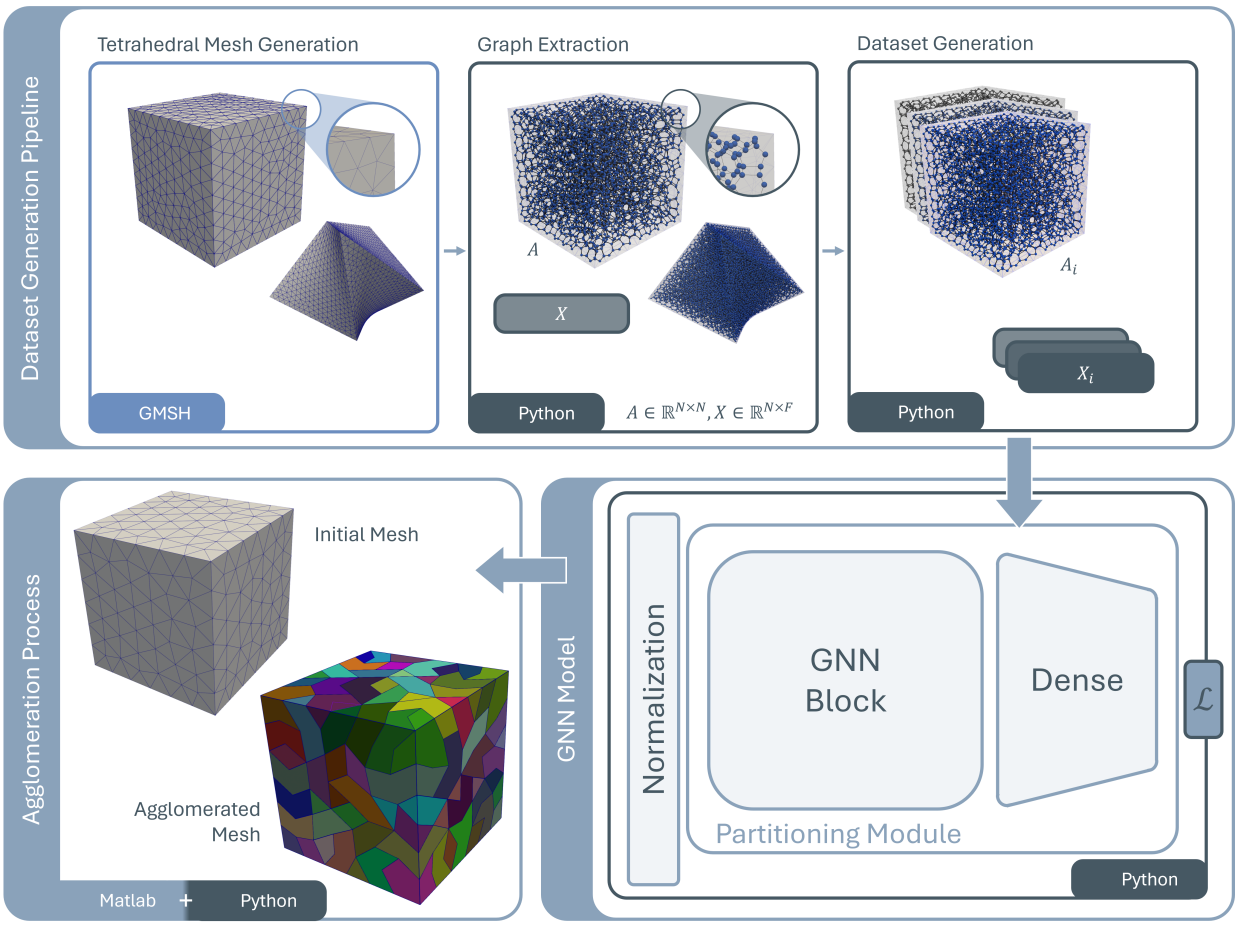


Figure 2: Schematic representation of the agglomeration pipeline

Notice that GNN-drive agglomeration techniques were proposed in [27], for a two-dimensional setting and homogeneous domain. However, the approach in [27] was limited to two-dimensional cases. The main novelty of the extension considered here is the possibility to handle also three-dimensional settings and properly take into account heterogeneities and/or embedded interfaces. More precisely, to handle the variability of the 3D meshes, which are much more complex than the 2D ones, we propose suitable improvements with respect to the algorithm proposed in [27]. Those improvements are both related to the pipeline and the GNN's architecture. Moreover, when considering a three-dimensional setting, training and inference need to be feasible with much larger datasets, due to the broader amount of elements that 3D meshes contain, even just in simple geometries. Moreover, we propose also a GNN architecture which is able to treat the agglomeration of heterogeneous domains, changing the loss function of the network.

In Figure 2 we report a schematic graphical conceptualization of the workflow of the proposed GNN agglomeration algorithm. In all cases the dataset of meshes is created by means of GMSH [48]. Then, using a Python code, the graph is extracted for each mesh and the dataset is created by bundling graphs adjacencies and features matrices) together. Then the GNN model is trained in an unsupervised learning setting, using a PyTorch Geometric [49] implementation. Finally, the trained model is loaded in evaluation mode and used to perform the graph bisection iteratively, obtaining a clustered graph, which is then post-processed in a Matlab environment to build the agglomerated mesh.

2.1 Architectures of GNN-agglomeration models for homogeneous domains

In this section, we present the proposed architecture of the GNN for mesh agglomeration for homogeneous domains. In particular, we explain the proposed architecture (GNN-AGGL-Enhanced), comparing it with the simplest possible

	GNN-AGGL-Base	GNN-AGGL-Enhanced
Normalization Layer	Normalization of the features in X: • Coordinates rescaled in $[-1, 1]$. • Volumes rescaled in $[0, 1]$. Rotation to align the maximum stretch direction with x-axis.	Normalization of the features in X: • Coordinates standardization: Mean: $\mu = 0$ - Variance: $\sigma^2 = 1$ • Volumes rescaled in $[0, 1]$. Rotation to align the maximum stretch direction with x-axis.
SAGEConv Layer	• 4 layers of 64 neurons. • Activation function: tanh.	• 4 layers of 128 neurons. • Activation function: tanh.
Linear Layer	3 decreasing-layers of $32 - 8 - 2$ neurons.	4 decreasing-layers of $64 - 32 - 8 - 2$ neurons.
SoftMax Layer	Single layer with softmax function to bisect the result into the two classes S_1 and S_2, to obtain the resulting matrix Y.	

Table 1: Main features of the two GNN architectures with a comparison of the specific features for each layer.

extension of the one proposed in [27] (GNN-AGGL-Base). Indeed, graph partitioning is a problem that focuses on the structure of the graph itself regardless of any specific spatial representation. This implies that for graphs in two or three dimensions, the principles and algorithms for graph partitioning remain the same, while obviously, suitable modifications to handle 3D data are needed. However, from the comparison of the result reported in Section 3 below, it is evident the need of a more complex GNN architecture, as the GNN-AGGL-Enhanced one, for the agglomeration task in 3D.

The proposed models (GNN-AGGL-Base and GNN-AGGL-Enhanced) are both based on two GNN SAGE-Base models, extending the ideas of [27], to the three-dimensional space. Both graph-bisection models perform a classification task taking as input the graph $G = (V, E)$ and the features related to its nodes $X \in \mathbb{R}^{N\times 4}$. The columns of the input feature matrix X at the i-th row correspond to the information of the i-th mesh element. In particular, we have:

$$X = \begin{bmatrix} x_1 & y_1 & z_1 & V_1 \\ \vdots & \vdots & \vdots & \vdots \\ x_i & y_i & z_i & V_i \\ \vdots & \vdots & \vdots & \vdots \\ x_N & y_N & z_N & V_N \end{bmatrix},$$

where (x_i, y_i, z_i) are the (x, y, z)-coordinates of the i-th element's centroid, while the V_i is the element's volume. The output is $Y \in \mathbb{R}^{N\times 2}$, where the first column represents the probability of belonging to the class S_1, while the second is the probability of being in the class S_2 of the partition. The characteristics of the resulting networks are reported in Table 1. We underline that the number of parameters of the GNN-AGGL-Enhanced model is $\sim$ 110K, with respect to the $\sim$ 28K of the GNN-AGGL-Base model. However, as will be depicted in Section 3, this level of complexity is required to correctly tame the inherited geometric complexity.

Training procedure

The training of the models has been performed in an unsupervised learning setting. The loss function that is used is the *expected normalized cut*. For details regarding the loss function construction, we refer to [27].

Concerning the construction of the dataset, we generate a training dataset of 500 tetrahedral meshes (125 meshes of the unitary cube and 375 meshes of suitable portions of them). All the meshes are generated using GMSH [48]. We adopt a split between the training set and validation set of respectively the 80% and the 20% of the total dataset. As a consequence, The training set is composed of 400 grids (100 meshes of the unitary cube and 300 meshes of portions), while the validation dataset is made by the remaining 100 grids. To guarantee good agglomeration results with our GNN models, we use a dataset with a large variety of meshes, differentiating the number of mesh elements. In the training and validation datasets, we start from coarse meshes of 600 elements, until around 42 000 elements of mesh in the finest one. In Figure 3, we report some examples of meshes from the training set.

Due to the low dimension of such datasets, we also adopt the two strategies to avoid overfitting effects:

- Regularization, summing an L^2-regularization term to the normalized cut loss, leading to the following loss

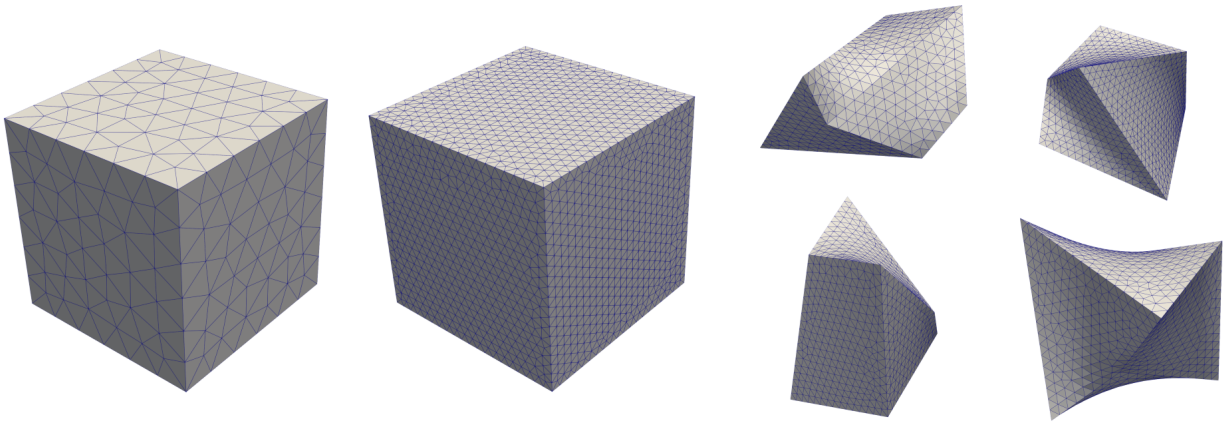


Figure 3: Examples of training dataset's meshes: meshes of a unitary cube (first two on the left) and a mesh of portions of the unitary cube (four ones on the right).

	GNN-AGGL-Base	**GNN-AGGL-Enhanced**
Number of epochs	300 Epochs	400 Epochs
Batch size	4 Samples	4 Samples
Learning rate (γ)	10^{-5}	10^{-4}
Weight-decay parameter (λ)	10^{-5}	10^{-5}

Table 2: Training hyper-parameters adopted for the two GNN proposed models.

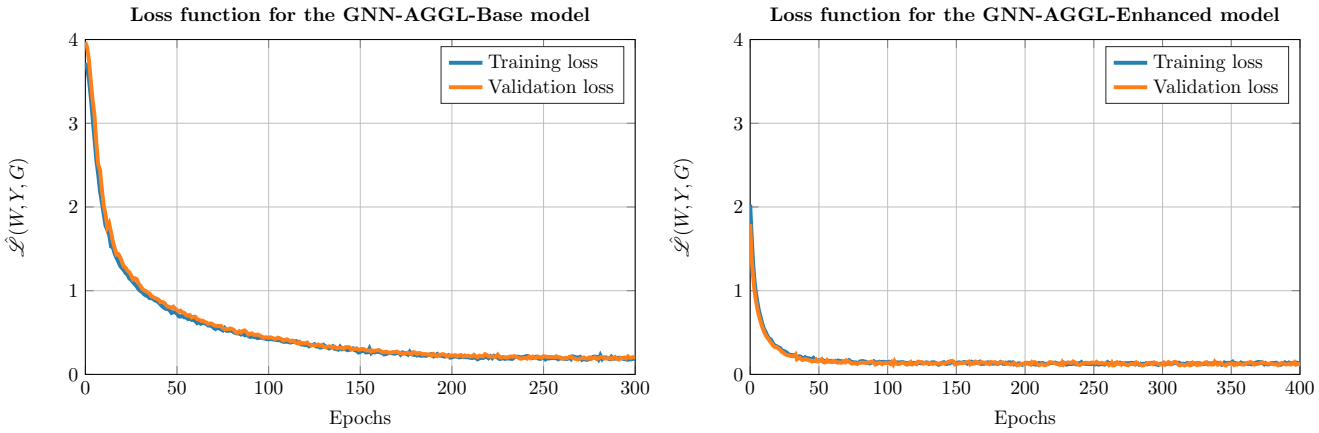


Figure 4: Training and validation loss functions for the GNN-AGGL-Base model (left) and the GNN-AGGL-Enhanced model (right).

definition:

$$\hat{\mathscr{L}}(W,Y,G) = \mathscr{L}(W,Y,G) + \lambda\|W\|_2^2.$$

- Data-augmentation, by implementing a rotation of each sample by a random angle at each training iteration. This approach makes the model more robust with respect to the input mesh configurations.

The training process has been performed by adopting an Adam optimizer. All the details about the training hyper-parameters for the two models are reported in Table 2.

In Figure 4, we report the values of the loss functions for the training and validation sets for the GNN-AGGL-Base and the GNN-AGGL-Enhanced models. We can observe that using the GNN-AGGL-Enhanced model, the loss function reaches lower values on both datasets. Moreover, the training procedure is more stable in contrast to what happens in the two-dimensional setting [27]. This is probably a consequence of the fact that the grids of the three-dimensional training dataset are made only of tetrahedra and not of general polyhedra.

2.2 Architectures of GNN-agglomeration model for heterogeneous domains

In this section, we further extend the proposed approach to perform mesh agglomeration on heterogeneous domains with different physical parameters. Specifically, this model can be exploited in different situations such as:

- Heterogeneous domains, where the main goal of the algorithm is to maintain as much as possible the distinction between the different regions still ensuring an acceptable shape for the agglomerated elements.
- Domains with micro-structures, which is a more complex situation because there is not a precise criterion to perform the agglomeration, and the algorithm aims to understand how to group the tetrahedra to obtain a better grid for numerical methods.

The state-of-the-art methods such as METIS of k-means are not able to perform agglomeration of meshes describing heterogeneous domains and/or containing embedded interfaces. The unique possibility of performing this task with such methods is to do the agglomeration separately on the different regions and then merge them. However, the merging process of separate meshes is complex and time-consuming. Our model automates the procedure by exploiting the abilities of GNNs. Moreover, concerning domains with micro-structures, there is no chance of agglomerating them with state-of-the-art methods.

The GNN model proposed for the heterogeneous case is an extension of the GNN-AGGL-Enhanced model previously discussed. The most relevant change in the GNN architecture is the addition of a new input feature representing the physical parameter ρ_i of the region to which the i-th element belongs. In this way we obtain an input feature matrix $X \in \mathbb{R}^{Nx5}$:

$$X = \begin{bmatrix} x_1 & y_1 & z_1 & V_1 & \rho_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_i & y_i & z_i & V_i & \rho_i \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_N & y_N & z_N & V_N & \rho_N \end{bmatrix},$$

where (x_i, y_i, z_i) are the (x, y, z)-coordinates of the i-th element's center, while the V_i is the element's volume. As explained for the other quantities, the parameter ρ_i would be re-scaled to obtain values between 0 and 1. This practice reduces the variability of the data, making it easier for the network to learn the relative importance of this parameter on the final result.

It is well known that neural networks often struggle to handle sharp changes in input data that may lead to gradient instability and hinder the model's generalization ability. Adding a new pre-processing operation can easily overcome this obstacle for heterogeneous domains. For this reason, we substitute in the normalization layer the last input feature with the average of such values in the adjacent elements. This step allows for avoiding sharp changes at the boundaries of distinct physical regions by smoothing out this transition.

The output of the GNN remains unchanged. It is $Y \in \mathbb{R}^{Nx2}$, where the first column represents the probability of belonging to S_1, while the second is the probability of being in S_2.

The idea behind the agglomeration algorithm is similar to that described in Algorithm 1. However, we introduce some modifications to better deal with the specific task of this model. Specifically, we insert an additional check before adding the agglomerated element to the final mesh to ensure the polytopal element connectivity. If it is not connected (an efficient algorithm for finding the connected components of a graph is reported in [50]) we perform an extra agglomeration step. The need to introduce this check is due to the network's tendency to keep elements with the same physical parameter together, also in cases when they are not connected.

Training procedure

Since this work is intended to be the first approach to GNN methods for heterogeneous mesh agglomeration, as a starting point we use a training dataset composed only of heterogeneous domains with a small number of parameters and regions. However, it is important to emphasize that this choice allows us to perform agglomeration also in the most general situations of computational domains with S parameters or domains with micro-structures, as shown in the forthcoming numerical tests. Moreover, this choice allows us to evaluate the generalization capabilities of the model.

The training dataset contains 200 tetrahedral meshes (100 with a single physical parameter and 100 with two heterogeneous parameters). The validation dataset contains 50 tetrahedral meshes (25 with a single physical parameter and 25 with two different parameter regions). The choice of the number of elements in the dataset

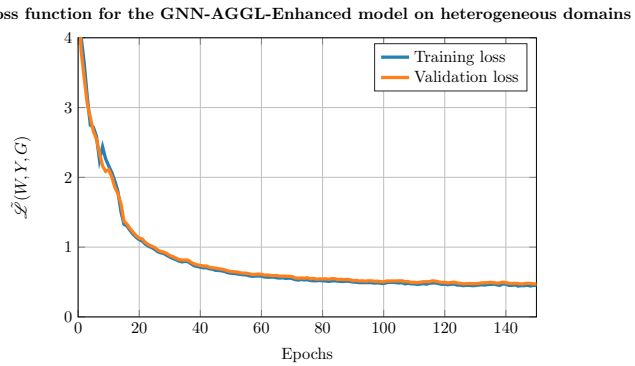


Figure 5: Training and validation loss functions for the GNN-AGGL-Enhanced model for heterogeneous domains.

preserves the 80/20 split between training and validation sets. The elements of the meshes in the two considered datasets are approximately 200 for the coarsest mesh and 17 000 for the finest one.

The training of the models has been performed in an unsupervised learning setting. We slightly modify the loss function by adding a further term to consider the physical feature. Hence, the new loss function includes a term involving the physical parameters. Consider the matrix $P \in \mathbb{R}^{N\times 2}$, that contains in the first column at the i-th row the "physical" parameter p_i and in the second one its complementary $1-p_i$ associated with the i-th mesh element. Then, the physical penalty part of the loss function reads:

$$\mathscr{P}(P,Y) = \sum P \odot Y, \tag{1}$$

where $\odot$ denotes the element-wise multiplication and the summation is over all the entries of the resulting matrix. Finally, the loss function of the GNN is composed of the sum of the terms in Equations (2.1) and (1):

$$\tilde{\mathscr{L}}(W,P,Y,G) = \alpha\mathscr{P}(P,Y) + \beta\left(\mathscr{L}(W,Y,G) + \lambda\|W\|_2^2\right). \tag{2}$$

The proposed loss function contains two parameters, α and β, to correctly balance the relevance of the normalized cut term and the one concerning the physical parameters. A hyperparameters tuning procedure has been used to balance the two terms. Indeed, by reducing the importance of the normalized cut, the model separates the physical classes correctly but we obtain low-quality agglomerated elements. Instead, by reducing the significance of the physical parameter term, we do not take into proper account the physical classes. The optimal values we found for that model are $\alpha = 1.28$ and $\beta = 0.00022$.

The training has been performed by adopting 150 epochs of the Adam optimizer with a batch size of 4. The learning rate is set to be $\gamma = 10^{-4}$, and the weight decay parameter is $\lambda = 10^{-5}$ and batch size 4. In Figure 5 we report the decay of the loss function $\tilde{\mathscr{L}}$ evaluated on training and validation datasets.

3 Numerical results: mesh agglomeration for homogeneous domains

In this section, after the introduction of some quality metrics for polyhedral meshes, we introduce a test dataset to compare the GNN model performance with the state-of-the-art methods (i.e. METIS and k-means). Moreover, we present an application to a mesh of human brain ventricles.

First, let us define the quality metrics we use in the next test cases. These quality metrics are employed also in [51, 27], but we adapt them to the three-dimensional setting:

- **Circle Ratio (CR)**: ratio between the radius of the largest contained sphere and the radius of the smallest sphere containing the polyhedron K:

$$\mathrm{CR}(K) = \frac{\max\limits_{B(r)\subset K}\{r\}}{\min\limits_{K\subset B(r)}\{r\}}, \tag{3}$$

  where $B(r)$ is a ball of radius r. We point out that inscribed spheres tangent to each of the polyhedron's faces and circumscribed ones touching each of the polyhedron's vertices might not exist for not convex polyhedra. For this reason, we instead use the largest sphere contained in the polyhedron as the inscribed one and the smallest sphere containing the polyhedron as the circumscribed one [51].

Agglomeration Method	Circle Ratio	Uniformity Factor	Volume Difference
GNN-AGGL-Base	0.1342	0.4053	0.7166
GNN-AGGL-Enhanced	0.1946	0.7676	0.2355
METIS	0.1799	0.7183	0.3517
k-Means	0.2662	0.7982	0.3270

Table 3: Test case 1: average values of the computed quality metrics (CR, UF, and VD) for the agglomerated grids reported in Figure 6, obtained with different agglomeration strategies (GNN-AGGL-Base, GNN-AGGL-Enhanced, METIS, k-Means).

- **Uniformity Factor (UF)**: the ratio between the diameter of an element K and the mesh size h:

$$\mathrm{UF}(K) = \frac{\mathrm{diam}(K)}{h}, \tag{4}$$

  where $h = \max_{K \in \mathscr{T}_h} \{\mathrm{diam}(K)\}$.

- **Volumes Difference (VD)**:

$$\mathrm{VD}(K) = \frac{|V(K) - V_{\mathrm{target}}|}{V_{\mathrm{target}}}, \tag{5}$$

  where $V_{\mathrm{target}} = \frac{\sum_{K \in \mathscr{T}_h} V(K)}{N_{\mathrm{el}}}$, where N_{el} is the number of the agglomerated elements of the mesh. Then, the target volume is the volume of an element in the agglomerated mesh if all the portions have the same volume.

The CR and UF quality metrics assume values between 0 and 1. Moreover, if the elements are regular, these quality metrics are close to 1. Finally, we introduce a metric to evaluate the performance of the agglomeration algorithm in terms of the difference between the volumes of the agglomerated mesh elements. VD assumes positive values and if the mesh is quasi-uniform, these values are close to 0.

3.1 Test case 1: agglomeration capabilities assessment of a test dataset

We construct a test dataset of 100 tetrahedral meshes of different refinements, with 563 elements for the coarsest mesh and 6 404 for the finest one. The dataset is composed as follows:

1. 30 meshes of the unitary cube,
2. 70 meshes of suitable portions of the unitary cube.

The computational domains are analogous to the ones used for training and we refer to Figure 3 that reports a mesh for each type. The results in the following sections are obtained using this test dataset. The considered agglomerated meshes have approximately 128 elements, corresponding to 7 bisection steps.

In Figure 6, we report a mesh from the test dataset together with the agglomerated meshes obtained using four different partitioning methods, namely GNN-AGGL-Base, GNN-AGGL-Enhanced, METIS, and k-means. We plot the agglomerated meshes with an "exploded" view for visualization purposes. This allows us to see how tetrahedra are agglomerated together into the new mesh elements. We can see that GNN-AGGL-Enhanced and k-means grids look quite similar and the agglomerated elements seem to be shape regular. On the contrary, METIS and GNN-AGGL-Base ones are of lower quality, due to the presence of stretched elements.

The following Table 3 collects the average values over the full test set of the quality metrics (CR, UF, and VD) for the agglomerated grids obtained employing different agglomeration strategies (GNN-AGGL-Base, GNN-AGGL-Enhanced, METIS, and k-Means). Moreover, in Figure 7, we report the violin plots relative to the distributions of the quality metrics, for the considered agglomerated mesh. Looking at the results, it is clear that the GNN-AGGL-Base model is not able to create an agglomerated mesh of sufficiently good quality. As expected, due to the much greater variability in the three-dimensional contest, the performance of this model is significantly lower than the one obtained in the 2D case [27]. For this reason, we focus only on the GNN-AGGL-Enhanced model in the following test cases.

On the contrary, we can see that the GNN-AGGL-Enhanced model performs better than METIS. This can be easily explained because the latter considers only the information coming from the graph topology while GNN-AGGL-Enhanced takes in input some geometrical properties of the mesh. Observing also k-means, we can infer

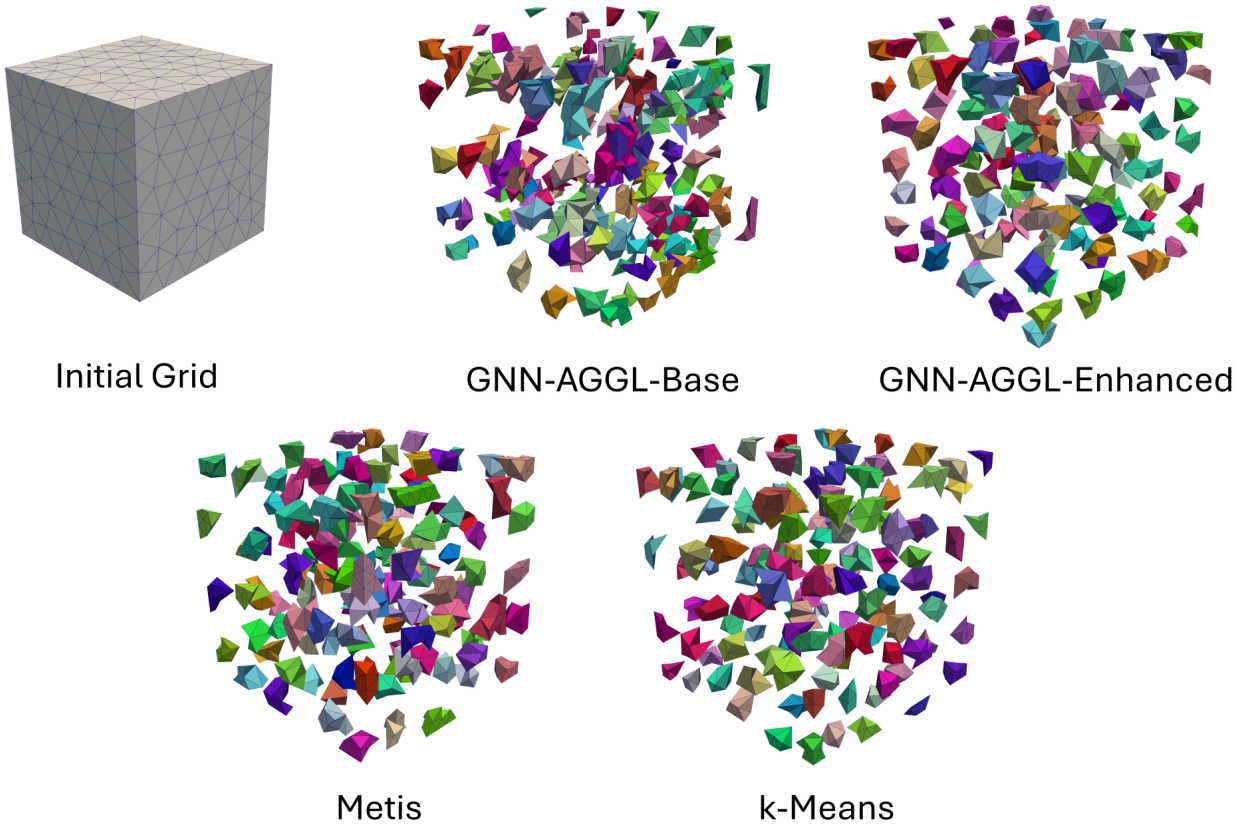


Figure 6: Test case 1: agglomerated grids obtained based on employing the GNN-AGGL-Base, the GNN-AGGL-Enhanced, METIS, and k-means algorithm.

that the developed algorithm is the best in terms of UF, while it is slightly worse than the performance of k-means in terms of shape regularity (CR index). However, we observe that the CR index provides low values for all the considered methods.

Runtime Performance

In Figure 8, we report the runtime performance of the proposed GNN-AGGL-Enhanced algorithm, compared with the state-of-the-art ones (METIS and k-Means). On the abscissa, we show the graph size, i.e. the number of meshes' elements, and on the ordinate, we plot the time required for the graph's bisection. The main advantage of our model with respect to the others is its non-iterative nature in performing graph bisection. The time benchmarking test set contains 15 tetrahedral meshes of increasing size, from 24 to 19 049 elements.

As expected, due to the above-mentioned advantage, the GNN-AGGL-Enhanced model outperforms both METIS and k-means in terms of runtime performance. Concerning the homogeneous domain agglomeration, the speed of the developed model is a key point in the discussion and allows us to consider the remarkable agglomeration capabilities (in terms of the final mesh's shape regularity) of our model slightly worse than, for example, k-means.

3.2 Test case 2: agglomeration of a mesh of human brain ventricles

To further test the generalization capabilities of the developed algorithm, we apply it to a more complex domain than the unitary cubes and portions of them considered so far. In particular, we consider some human brain ventricles reconstructed starting from a Magnetic Resonance Imaging (MRI) scan taken from the OASIS database [52]. Brain ventricles are a network of interconnected cavities filled with cerebrospinal fluid. The domain is highly non-convex and presents many convoluted pathways, constrictions, and narrowed sections. However, due to the actual research on brain fluid dynamics, constructing a high-quality polyhedral mesh is of major interest [2, 53].

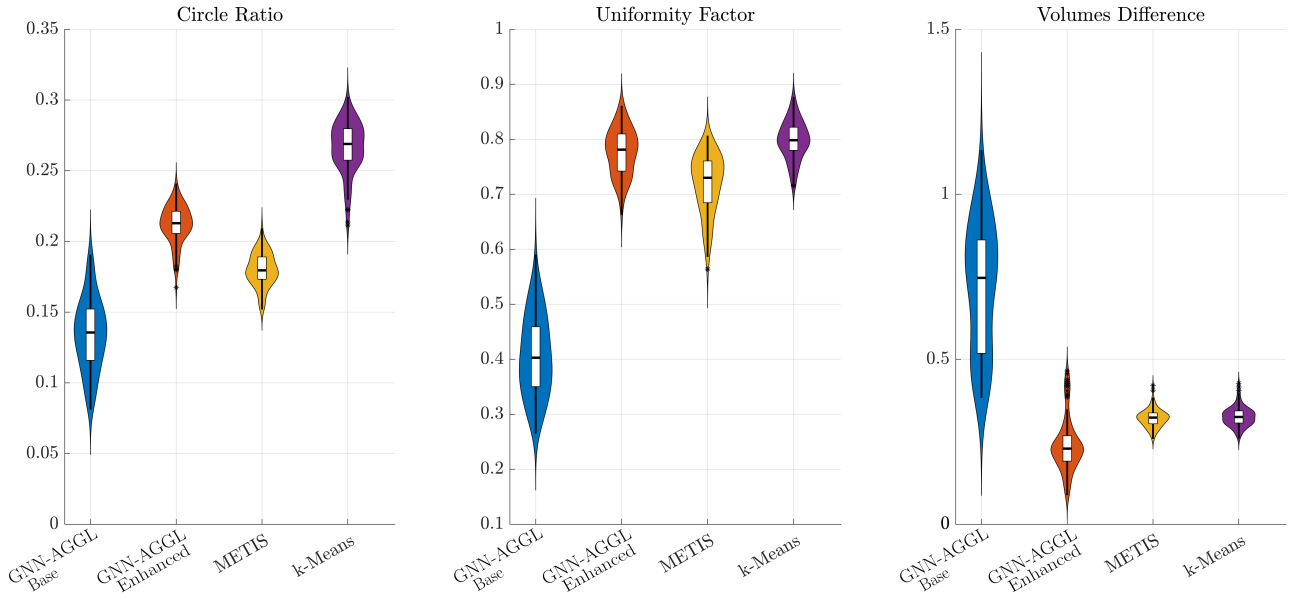


Figure 7: Test case 1: violin plots of the computed quality metrics (CR on the left, UF in the middle, and VD on the right) for the agglomerated grids reported in Figures 6, obtained with different agglomeration strategies (GNN-AGGL-Base, GNN-AGGL-Enhanced, METIS, k-Means).

The starting mesh consists of 28 706 tetrahedral elements, as shown in Figure 9.

We agglomerated such a grid using GNN-AGGL-Enhanced, METIS, and k-means algorithms. The target mesh size of Algorithm 1 used for this test case is $h/10$, where h is the diameter of the original mesh. In this way, we obtain three agglomerated meshes with approximately 300 elements. In Figure 9 we report the results of the agglomeration processes, also considering, a detail of the agglomerated mesh of the fourth ventricle and aqueduct of Sylvius.

Agglomeration Method	**Circle Ratio**	**Uniformity Factor**
GNN-AGGL-Enhanced	0.2252	0.6393
METIS	0.2227	0.6151
k-Means	0.2313	0.6261

Table 4: Test case 2: average values of the computed quality metrics (CR and UF) for the agglomerated grids reported in Figure 9, obtained with different agglomeration strategies (GNN-AGGL-Enhanced, METIS, k-Means).

In Table 4 we report the average values of the quality metrics (CR and UF) computed on the agglomerated values. The VD metric is not reported because the starting mesh and as a consequence the agglomerated ones are not uniform due to the geometrical conformation of the domain. Due to this fact, the evaluation of the uniformity of the volumes of the elements is not significant for the test case. The three considered algorithms show similar results. This confirms that the proposed GNN-AGGL-Enhanced algorithm is as accurate as the state-of-the-art ones. Moreover, it indicates a good generalization capability of the GNN-AGGL-Enhanced model, considering that the computational domain is totally different from the ones included in the training set, both in terms of shape and dimensions.

4 Numerical results: mesh agglomeration for heterogeneous domains

In this section, after the introduction of an additional quality metric to analyze the agglomeration domains with heterogeneous physical parameters, we introduce some test datasets to evaluate the GNN-AGGL-Enhanced model. Moreover, we present two applications, the agglomeration of a domain containing microstructure and of a human brain.

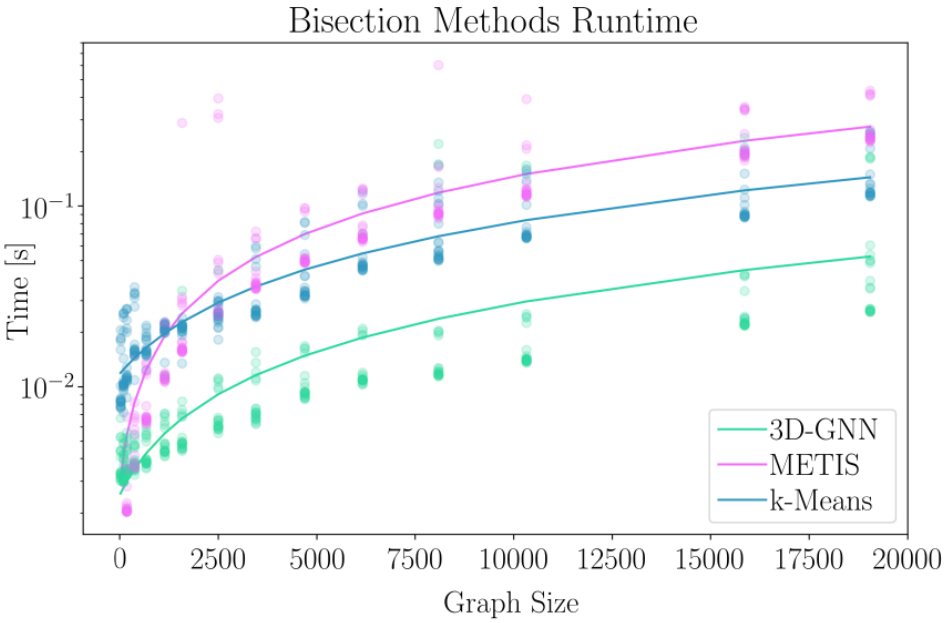


Figure 8: Test case 1: runtime performance for different graph bisection models (METIS, k-Means, and GNN-AGGL-Enhanced) as a function of the number of nodes in the connectivity graph of tetrahedral meshes.

To evaluate the shape regularity of the elements in the agglomerated meshes, we use the quality metrics introduced in Section 3. However, we need to introduce a new quantity, that will be later used to measure the ability of the algorithm to preserve the different physical regions separated in the agglomeration process. We define the **Heterogeneous Elements (HE)** as the percentage of elements in the agglomerated mesh that contain discontinuities in the physical parameters.

$$\text{HE} = \frac{\#\text{heterogeneous elements}}{\#\text{elements}}\%. \tag{6}$$

The quantity should take values close to 0 as much as possible, to avoid the presence of agglomerated elements containing discontinuities in the physical parameters of the domain.

4.1 Test case 3: agglomeration capabilities assessment of test datasets

In this test case, we consider three different test datasets, constructed on unitary cubes, but with different distribution of physical parameters. In particular, the datasets are composed as follows:

- **Dataset 1**: 25 tetrahedral meshes with a number of elements between 373 and 10 389 with the same physical parameter in the whole domain.
- **Dataset 2**: 25 tetrahedral meshes with a number of elements between 437 and 8 549 with two different physical parameters in two regions of the domain.
- **Dataset 3**: 50 tetrahedral meshes with a number of elements between 529 and 14 960 with two different physical parameters in more than two regions of the domain.

As an example, we report a random mesh for each of the presented datasets in Figure 10. Different colours are used to represent a different value of the physical parameter and the bold lines denote the region delimitations.

The results in this section are obtained using these datasets and clustering until reaching a diameter of the agglomerated elements of 0.25 times the diameter of the original mesh h. In Figure 10, we report an example of mesh for each of the three test sets, together with the agglomerated meshes. The agglomerated meshes are visualized separating the different physical parameter regions, to underline the good level of separation of heterogeneities obtained by the GNN algorithm. In particular, only the example in Dataset 3 shows a single element containing two different physical values in the middle region. Finally, we can see that the agglomerated elements seem to have good shapes and to be quite uniform in all the considered cases.

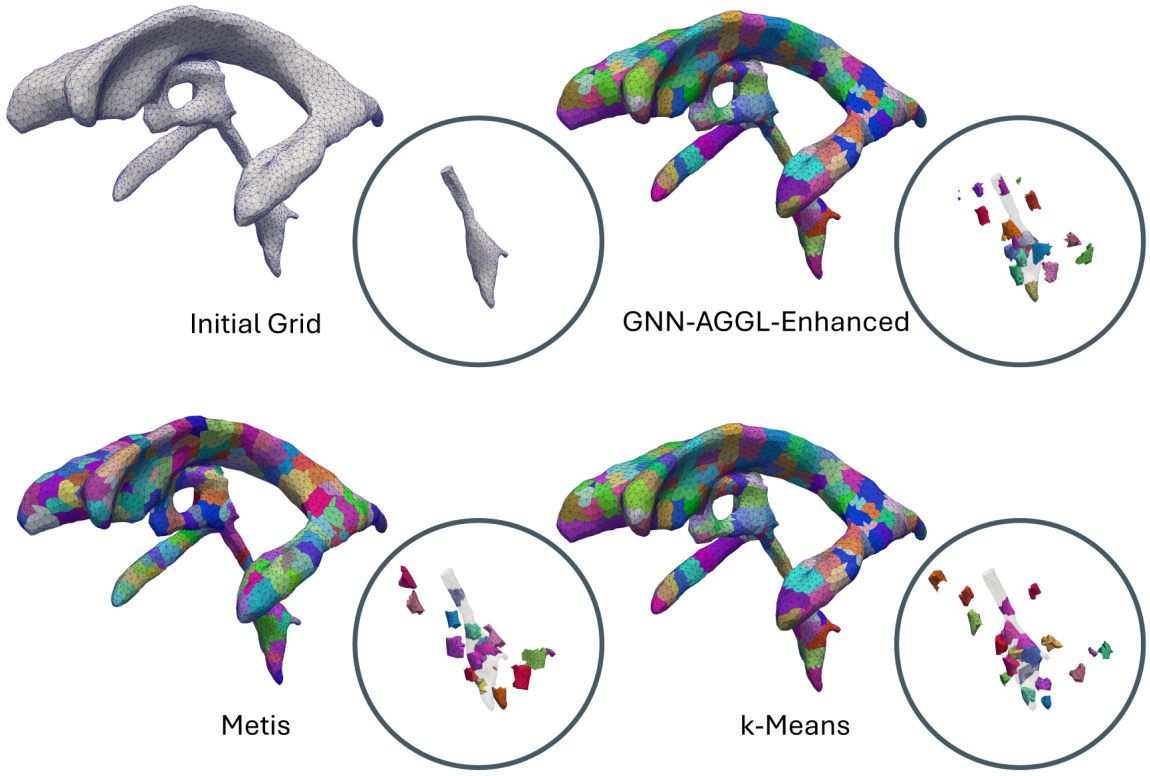


Figure 9: Test case 2: initial mesh of the human brain's ventricles and agglomerated meshes using different strategies (GNN-AGGL-Enhanced, METIS, and k-means). In the circles, we depict a detail of the exploded agglomerated grids of the fourth ventricle and aqueduct of Sylvius.

Dataset	CR	UF	VD	HE
Dataset 1	0.1962	0.7568	0.3390	-
Dataset 2	0.1918	0.7592	0.3692	1.8186 %
Dataset 3	0.1996	0.7362	0.5075	2.3384 %

Table 5: Test case 3: average values of the computed quality metrics (CR, UF, VD, and HE) for the agglomerated grids reported in Figure 10.

To have a quantitative understanding of the agglomeration quality, we collect the average values of the quality metrics in the three different datasets in Table 5. First of all, the quantitative feedback confirms the good behavior of such a model in agglomerating while preserving the different physical regions. Indeed, on average, approximately only 2% of the final elements is heterogeneous with respect to the physical parameters in the starting tetrahedral elements. This achievement is significant because it allows us to agglomerate the heterogeneous meshes with the awareness of obtaining a meaningful coarse mesh of the computational domain.

Moreover, the obtained values are coherent to what was obtained in the homogeneous case, with the values reported in Table 3. The important remark is that we do not make comparisons with the state-of-the-art methods because they are not able to perform automatically the agglomeration considering the heterogeneity in the physical parameter values. Therefore, the real strong point of this model is not the performance in terms of quality metrics, which can undoubtedly be improved by further developments, but the possibility of making this agglomeration process automatic, without a segregated agglomeration and a glue of the obtained meshes.

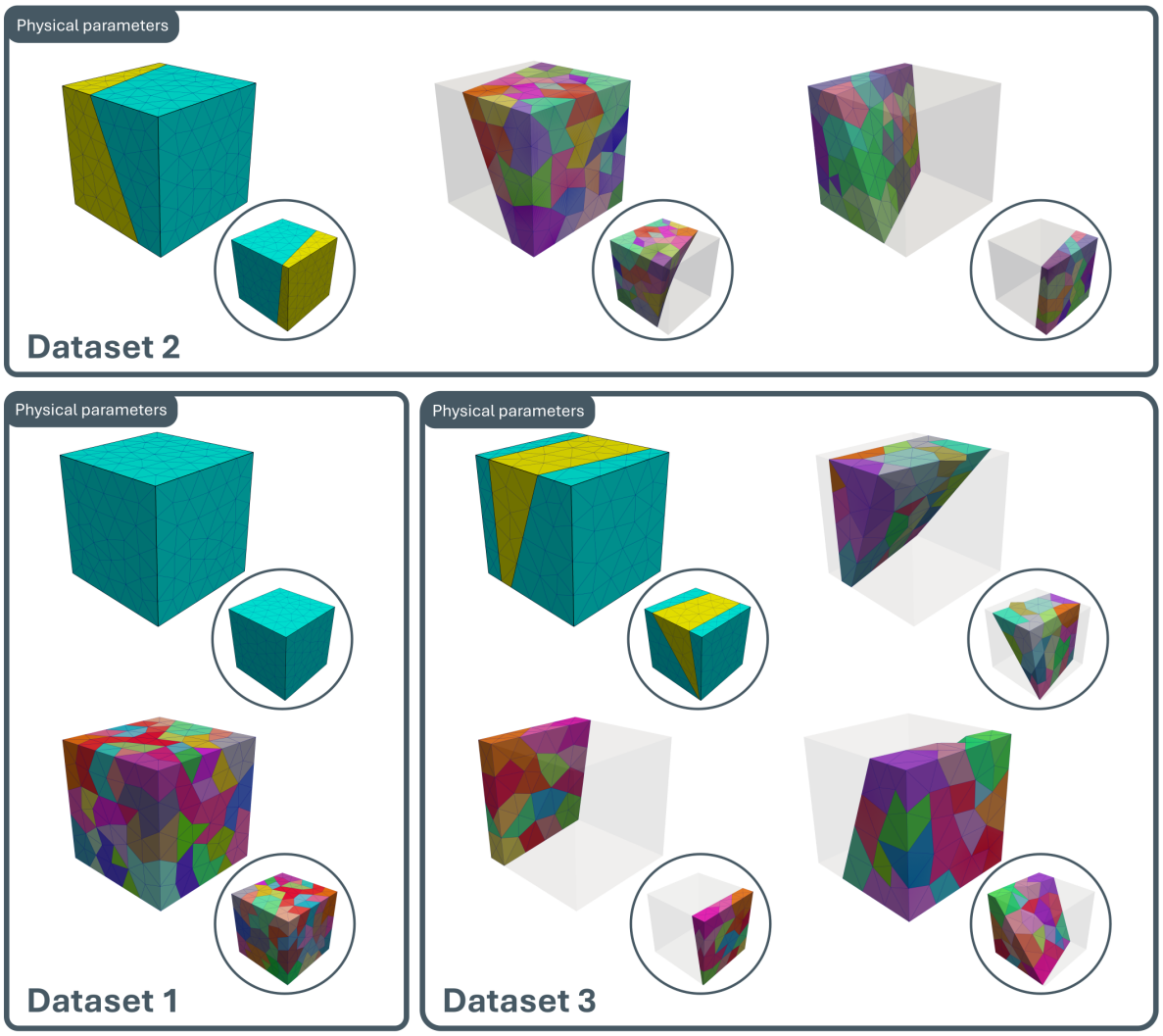


Figure 10: Test case 3: examples of agglomerated meshes, one for each dataset. The first one shown is always representing the physical parameter heterogeneities inside the domain and the others are the agglomerated elements grouped by main parameter value inside the element. Each figure is reported from two different views to better visualize the agglomerated elements.

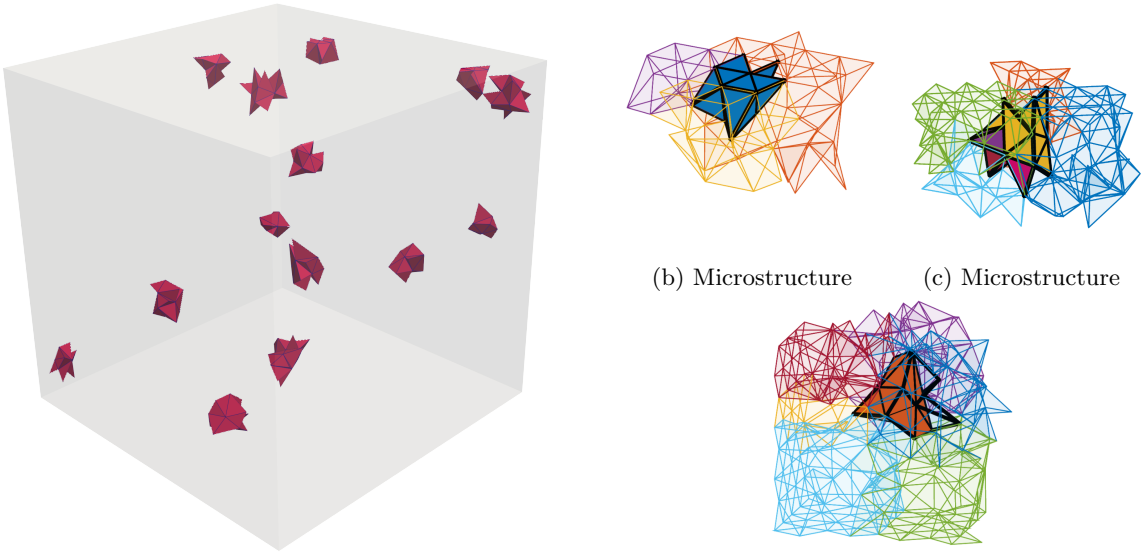
4.2 Test case 4: agglomeration of a mesh containing microstructures

Computational domains with complex microstructures at different scales are typical in the field of Integrated Computational Material Engineering (ICME). More specifically, many heterogeneous materials have this configuration, such as alloys, polymers, tissues, bones, and porous rocks [54, 55, 56, 57, 58].

The computational domain is a unitary cube with 15 microstructures in random positions, meshed by a tetrahedral grid made by 71 836 tetrahedral elements (see Figure 11a). Each microstructure is composed of 10 to 20 tetrahedra, and so it is of negligible dimensions with respect to the diameter of the domain. We model this domain by setting the "physical" parameter of the background to 0 and the microstructure one equal to 1.

We perform a mesh agglomeration process using the proposed GNN algorithm until reaching a diameter of the agglomerated mesh's elements less than $0.15\,h$, where h is the diameter of the original mesh. We get an agglomerated mesh of 2 264 elements. The first result to be underlined is the fact that our model, despite the microstructures having a small size, can identify each one of them. Indeed, where we have one of these particles in the initial grid, we found at least one agglomerated element with the physical parameter 1 in the final mesh.

In Figure 11, each shadowed color represents a different agglomerated element, the brighter colors are the agglomerates with physical parameter 1, and, finally, the bold black lines represent the boundary of the microstructure.



(a) Domain with microstructures

(b) Microstructure

(c) Microstructure

(d) Microstructure

Figure 11: Test case 4: domain with microstructure (a), and visual inspection of three domain's regions containing a microstructure (b),(c), and (d).

We observe that the model can gathers the tetrahedra of a microstructure in one agglomerated element as in Figure 11b. In some cases, the elements of one structure are divided into smaller agglomerates (see Figure 11c). In some instances, we find some tetrahedra of a microstructure inserted in one of the adjacent background's bigger elements as in Figure 11d. However, by calculating the percentage of tetrahedra of each microstructure correctly classified with physical parameter 1 in the final grid, we obtain a value around 84.90%.

4.3 Test case 5: agglomeration of a mesh of a human brain

The final test proposed in this work is the application to a complex and heterogeneous domain. The brain is an intricated geometrical structure, with a natural distinction into two parts, namely, white and grey matter (see Figure 12). The computational domain can be reconstructed starting from MRI scans (in this work we consider one from the OASIS database [52]). The domain is also highly non-convex, and constructing efficient and high-quality tetrahedral meshes is complex. In particular, some applications require considering the heterogeneities in the cerebral medium for a correct physical description of the phenomenon [3, 59].

The computational domain is meshed by a tetrahedral grid made of 67 164 elements, and the grid is constructed to preserve the distinction between white and grey matter. We model the domain by setting the grey matter tag to 0 and the white matter to 1. We perform a mesh agglomeration process using the proposed GNN algorithm until reaching a diameter of the agglomerated mesh's elements less than $0.15\,h$, where h is the diameter of the original mesh. Finally, we get an agglomerated mesh of 3 230 elements.

In Figure 12, each shadowed color represents a different agglomerated element. We can observe a good distinction in the agglomeration with respect to the physical parameter describing the matter properties. A quantitative analysis confirms this quality. Indeed, by measuring the number of the tetrahedra agglomerated into an element with the wrong physical parameter, we can observe they are only the 2.0979% of the total. Moreover, considering that almost the 35.4059% of the initial mesh elements have a face on the interface between white and grey matters, we notice that only the 5.9253% of the latter group is agglomerated incorrectly. These results are of particular significance, considering that none of the state-of-the-art methods can address the task automatically and that the brain geometry and the internal interface are highly complex.

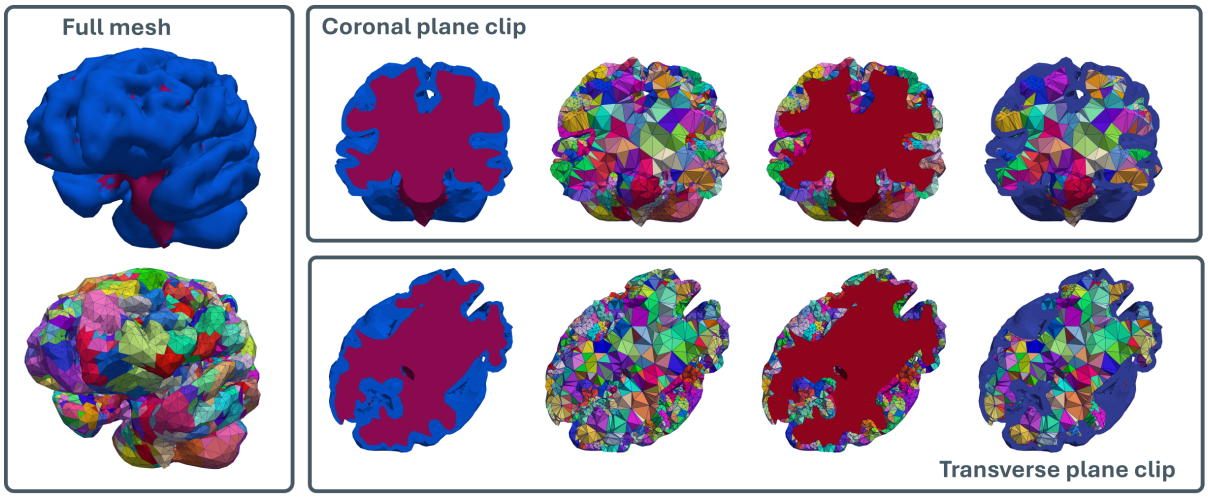


Figure 12: Test case 5: brain geometry with a distinction between white (in red) and grey (in blue) matters and agglomerated mesh. In the mesh visualization, the agglomerated elements are shown with different colors. We report a full brain visualization, a clipped view in the coronal plane, and one in the transverse plane. In the clipped views we also show separately the two matters agglomerated elements.

5 Conclusions

In this work, we have developed a novel agglomeration algorithm for three-dimensional grids, via a graph partitioning algorithm based on Graph Neural Networks. The model is based on learning the geometrical information of the mesh's elements automatically and cost-effectively, remaining independent of geometry and underlying differential models. This structure allows us to approach a wide range of situations, which would not be possible using classical strategies. Indeed, the proposed GNNs-enhanced algorithm can perform completely automatically the agglomeration of heterogeneous domains and ones containing microstructure, which is not possible by relying on classical methods. Another novelty of the proposed method consists of overcoming the main limitations of state-of-the-art approaches like METIS and k-means using GNN architectures, preserving the geometrical quality of the initial grid, and reducing the mesh complexity in terms of memory storage and associated computational costs.

First of all, we re-framed the problem of mesh agglomeration as a graph partitioning problem by exploiting the graph representation of the connectivity structure of the mesh's elements. Then, we developed in this context a framework to employ GNN-based strategies that can exploit the geometrical information of the grid. After, we compared the performance of the proposed GNN strategies (properly trained) with k-means and METIS, classical algorithms for graph partitioning, over a set of different grids, featuring also complex real domains such as brain ventricles from MRI scans. Results show that the GNN proposed method can preserve a high mesh quality, with the advantage of processing naturally and simultaneously both the graph structure of the mesh and the geometrical information. Moreover, GNNs have a significantly lower online computational cost.

Finally, we refined the model to perform the agglomeration of a heterogeneous domain and tested it on different types of heterogeneous datasets, to confirm the quality of the constructed agglomerated meshes. Finally, we test the GNN algorithm on the micro-structures framework and a complex domain such as a real brain geometry. We remark also that the state-of-the-art methods cannot automatically perform the agglomeration of these domains.

Future developments certainly include improving the generalization capabilities of the GNNs, by generating a larger training dataset. Another possibility is to re-frame the approach in a reinforcement learning framework, as proposed by [60].

Acknowledgments

The brain MRI images were provided by OASIS-3: Longitudinal Multimodal Neuroimaging: Principal Investigators: T. Benzinger, D. Marcus, J. Morris; NIH P30 AG066444, P50 AG00561, P30 NS09857781, P01 AG026276, P01 AG003991, R01 AG043434, UL1 TR000448, R01 EB009352. AV-45 doses were provided by Avid Radiopharmaceuticals, a wholly-owned subsidiary of Eli Lilly.

Declaration of competing interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this article.

References

[1] S. Zonca, P. F. Antonietti, and C. Vergara, "A polygonal discontinuous Galerkin formulation for contact mechanics in fluid-structure interaction problems", *Communications in Computational Physics*, vol. 30, no. 1, pp. 1–33, 2022.

[2] M. Corti, P. F. Antonietti, L. Dede', and A. M. Quarteroni, "Numerical modeling of the brain poromechanics by high-order discontinuous Galerkin methods", *Mathematical Models and Methods in Applied Sciences*, vol. 33, no. 08, pp. 1577–1609, 2023.

[3] M. Corti, F. Bonizzoni, L. Dede', A. M. Quarteroni, and P. F. Antonietti, "Discontinuous Galerkin methods for Fisher–Kolmogorov equation with application to α-synuclein spreading in Parkinson's disease", *Computer Methods in Applied Mechanics and Engineering*, vol. 417, p. 116450, 2023.

[4] P. F. Antonietti, S. Giani, and P. Houston, "*hp*-version composite discontinuous Galerkin methods for elliptic problems on complicated domains", *SIAM Journal on Scientific Computing*, vol. 35, no. 3, pp. A1417–A1439, 2013.

[5] A. Cangiani, E. H. Georgoulis, and P. Houston, "*hp*-version discontinuous Galerkin methods on polygonal and polyhedral meshes", *Mathematical Models and Methods in Applied Sciences*, vol. 24, no. 10, pp. 2009–2041, 2014.

[6] P. F. Antonietti, A. Cangiani, J. Collis, Z. Dong, E. H. Georgoulis, S. Giani, and P. Houston, "Review of Discontinuous Galerkin Finite Element Methods for Partial Differential Equations on Complicated Domains", in *Building Bridges: Connections and Challenges in Modern Approaches to Numerical Partial Differential Equations*, pp. 281–310, Springer, 2016.

[7] A. Cangiani, Z. Dong, E. H. Georgoulis, and P. Houston, *hp-Version Discontinuous Galerkin Methods on Polygonal and Polyhedral Meshes*. Springer International Publishing, 2017.

[8] P. F. Antonietti, C. Facciolà, P. Houston, I. Mazzieri, G. Pennesi, and M. Verani, "High–order Discontinuous Galerkin Methods on Polyhedral Grids for Geophysical Applications: Seismic Wave Propagation and Fractured Reservoir Simulations", in *Polyhedral Methods in Geosciences*, pp. 159–225, Springer International Publishing, 2021.

[9] P. Antonietti, S. Bonetti, M. Botti, M. Corti, I. Fumagalli, and I. Mazzieri, "lymph: discontinuous polytopal methods for multi-physics differential problems", 2024, ArXiv.

[10] L. Beirão Da Veiga, F. Brezzi, A. Cangiani, G. Manzini, L. D. Marini, and A. Russo, "Basic principles of virtual element methods", *Mathematical Models and Methods in Applied Sciences*, vol. 23, no. 01, pp. 199–214, 2013.

[11] L. Beirão da Veiga, F. Brezzi, L. D. Marini, and A. Russo, "The Hitchhiker's guide to the Virtual Element Method", *Mathematical Models and Methods in Applied Sciences*, vol. 24, no. 08, pp. 1541–1573, 2014.

[12] L. Beirão da Veiga, F. Brezzi, L. D. Marini, and A. Russo, "Virtual Element Method for general second-order elliptic problems on polygonal meshes", *Mathematical Models and Methods in Applied Sciences*, vol. 26, no. 04, pp. 729–750, 2016.

[13] L. B. d. Veiga, F. Brezzi, L. D. Marini, and A. Russo, "Mixed virtual element methods for general second order elliptic problems on polygonal meshes", *ESAIM: Mathematical Modelling and Numerical Analysis*, vol. 50, no. 3, pp. 727–747, 2016.

[14] P. F. Antonietti, L. Beirão Da Veiga, and G. Manzini, *The Virtual Element Method and its Applications*. Springer International Publishing, 2022.

[15] D. A. Di Pietro, A. Ern, and S. Lemaire, "An arbitrary-order and compact-stencil discretization of diffusion on general meshes based on local reconstruction operators", *Computational Methods in Applied Mathematics*, vol. 14, no. 4, pp. 461–472, 2014.

[16] D. A. Di Pietro and A. Ern, "Hybrid high-order methods for variable-diffusion problems on general meshes", *Comptes Rendus Mathematique*, vol. 353, no. 1, pp. 31–34, 2015.

[17] D. A. Di Pietro, A. Ern, and S. Lemaire, "A review of Hybrid High-Order methods: Formulations, computational aspects, comparison with other methods", in *Building Bridges: Connections and Challenges in Modern Approaches to Numerical Partial Differential Equations*, pp. 205–236, Springer International Publishing, 2016.

[18] D. A. Di Pietro and J. Droniou, *The Hybrid High-Order Method for Polytopal Meshes: Design, Analysis, and Applications*. Springer International Publishing, 2020.

[19] J. Hyman, M. Shashkov, and S. Steinberg, "The numerical solution of diffusion problems in strongly heterogeneous non-isotropic materials", *Journal of Computational Physics*, vol. 132, no. 1, pp. 130–148, 1997.

[20] F. Brezzi, K. Lipnikov, and V. Simoncini, "A family of mimetic finite difference methods on polygonal and polyhedral meshes", *Mathematical Models and Methods in Applied Sciences*, vol. 15, no. 10, pp. 1533–1551, 2005.

[21] F. Brezzi, K. Lipnikov, and M. Shashkov, "Convergence of the mimetic finite difference method for diffusion problems on polyhedral meshes", *SIAM Journal on Numerical Analysis*, vol. 43, no. 5, pp. 1872–1896, 2005.

[22] L. B. Da Veiga, K. Lipnikov, and G. Manzini, *The Mimetic Finite Difference Method for Elliptic Problems*. Springer International Publishing, 2014.

[23] B. Cockburn, B. Dong, and J. Guzmán, "A superconvergent LDG-hybridizable Galerkin method for second-order elliptic problems", *Mathematics of Computation*, vol. 77, pp. 1887–1916, 2008.

[24] B. Cockburn, J. Guzmán, and H. Wang, "Superconvergent discontinuous Galerkin methods for second-order elliptic problems", *Mathematics of Computation*, vol. 78, pp. 1–24, 2009.

[25] B. Cockburn, J. Gopalakrishnan, and R. Lazarov, "Unified hybridization of discontinuous Galerkin, mixed, and continuous Galerkin methods for second order elliptic problems", *SIAM Journal on Numerical Analysis*, vol. 47, no. 2, pp. 1319–1365, 2009.

[26] B. Cockburn, J. Gopalakrishnan, and F. J. Sayas, "A projection-based error analysis of HDG methods", *Mathematics of Computation*, vol. 79, pp. 1351–1367, 2010.

[27] P. Antonietti, N. Farenga, E. Manuzzi, G. Martinelli, and L. Saverio, "Agglomeration of polygonal grids using graph neural networks with applications to multigrid solvers", *Computers & Mathematics with Applications*, vol. 154, pp. 45–57, 2024.

[28] M. Feder, A. Cangiani, and L. Heltai, "R3MG: R-tree based agglomeration of polytopal grids with applications to multilevel methods", 2024, ArXiv.

[29] T. F. Chan and J. Zou, "A convergence theory of multilevel additive Schwarz methods on unstructured meshes", *Numerical Algorithms*, vol. 13, no. 2, pp. 365–398, 1996.

[30] F. Bassi, L. Botti, A. Colombo, D. Di Pietro, and P. Tesini, "On the flexibility of agglomeration based physical space discontinuous Galerkin discretizations", *Journal of Computational Physics*, vol. 231, no. 1, pp. 45–65, 2012.

[31] F. Bassi, L. Botti, A. Colombo, and S. Rebay, "Agglomeration based discontinuous Galerkin discretization of the Euler and Navier-Stokes equations", *Computers and Fluids*, vol. 61, pp. 77–85, 2012.

[32] P. F. Antonietti, M. Sarti, and M. Verani, "Multigrid algorithms for *hp*-discontinuous Galerkin discretizations of elliptic problems", *SIAM Journal on Numerical Analysis*, vol. 53, no. 1, pp. 598–618, 2015.

[33] P. Antonietti, P. Houston, X. Hu, M. Sarti, and M. Verani, "Multigrid algorithms for *hp*-version interior penalty discontinuous Galerkin methods on polygonal and polyhedral meshes", *Calcolo*, vol. 54, no. 4, pp. 1169–1198, 2017.

[34] J. Xu and L. Zikatanov, "Algebraic multigrid methods", *Acta Numerica*, vol. 26, pp. 591–721, 2017.

[35] P. Antonietti and G. Pennesi, "V-cycle multigrid algorithms for discontinuous Galerkin methods on non-nested polytopic meshes", *Journal of Scientific Computing*, vol. 78, pp. 625–652, 2019.

[36] M. Raissi and G. E. Karniadakis, "Hidden physics models: Machine learning of nonlinear partial differential equations", *Journal of Computational Physics*, vol. 357, pp. 125–141, 2018.

[37] J. S. Hesthaven and S. Ubbiali, "Non-intrusive reduced order modeling of nonlinear problems using Neural Networks", *Journal of Computational Physics*, vol. 363, pp. 55–78, 2018.

[38] D. Ray and J. S. Hesthaven, "An artificial neural network as a troubled-cell indicator", *Journal of Computational Physics*, vol. 367, pp. 166–191, 2018.

[39] M. Raissi, P. Perdikaris, and G. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations", *Journal of Computational Physics*, vol. 378, pp. 686–707, 2019.

[40] F. Regazzoni, L. Dede', and A. M. Quarteroni, "Machine learning for fast and reliable solution of time-dependent differential equations", *Journal of Computational Physics*, vol. 397, 2019.

[41] F. Regazzoni, L. Dede', and A. M. Quarteroni, "Machine learning of multiscale active force generation models for the efficient simulation of cardiac electromechanics", *Computer Methods in Applied Mechanics and Engineering*, vol. 370, p. 113268, 2020.

[42] F. Regazzoni, M. Salvador, L. Dede', and A. Quarteroni, "A machine learning method for real-time numerical simulations of cardiac electromechanics", *Computer Methods in Applied Mechanics and Engineering*, vol. 393, p. 114825, 2022.

[43] P. F. Antonietti, M. Caldana, and L. Dede', "Accelerating algebraic multigrid methods via artificial neural networks", *Vietnam Journal of Mathematics*, vol. 51, no. 1, pp. 1–36, 2023.

[44] P. F. Antonietti, F. Dassi, and E. Manuzzi, "Machine learning based refinement strategies for polyhedral grids with applications to Virtual Element and polyhedral discontinuous Galerkin methods", *Journal of Computational Physics*, p. 111531, 2022.

[45] P. F. Antonietti and E. Manuzzi, "Refinement of polygonal grids using Convolutional Neural Networks with applications to polygonal discontinuous Galerkin and Virtual Element methods", *Journal of Computational Physics*, vol. 452, p. 110900, 2022.

[46] J. MacQueen, "Some methods for classification and analysis of multivariate observations", in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pp. 281–298, University of California Press, 1967.

[47] G. Karypis and V. Kumar, "A fast and high quality multilevel scheme for partitioning irregular graphs", *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 359–392, 1998.

[48] C. Geuzaine and J. F. Remacle, "GMSH: A 3-D Finite Element mesh generator with built-in pre- and post-processing facilities", *International Journal for Numerical Methods in Engineering*, vol. 79, no. 11, pp. 1309–1331, 2009.

[49] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library", in *Advances in Neural Information Processing Systems 32*, pp. 8024–8035, Curran Associates, Inc., 2019.

[50] R. Tarjan, "Depth-first search and linear graph algorithms", *SIAM journal on Computing*, vol. 1, no. 2, pp. 146–160, 1972.

[51] M. Attene, S. Biasotti, S. Bertoluzza, D. Cabiddu, M. Livesu, G. Patanè, M. Pennacchio, D. Prada, and M. Spagnuolo, "Benchmarking the geometrical robustness of a Virtual Element Poisson solver", *Mathematics and Computers in Simulation*, vol. 190, pp. 1392–1414, 2021.

[52] P. J. LaMontagne, T. L. Benzinger, J. C. Morris, S. Keefe, R. Hornbeck, C. Xiong, E. Grant, J. Hassenstab, K. Moulder, A. G. Vlassenko, M. E. Raichle, C. Cruchaga, and D. Marcus, "OASIS-3: Longitudinal neuroimaging, clinical, and cognitive dataset for normal aging and Alzheimer disease". medRxiv, 2019.

[53] I. Fumagalli, M. Corti, N. Parolini, and P. Antonietti, "Polytopal discontinuous Galerkin discretization of brain multiphysics flow dynamics", *Journal of Computational Physics*, vol. in press, p. 113115, 2021.

[54] Y. Gao, Y. Jiao, and Y. Liu, "Ultra-efficient reconstruction of 3D microstructure and distribution of properties of random heterogeneous materials containing multiple phases", *Acta Materialia*, vol. 204, p. 116526, 2021.

[55] Z. Trivedi, D. Gehweiler, J. K. Wychowaniec, T. Ricken, B. Gueorguiev, A. Wagner, and O. Röhrle, "A continuum mechanical porous media model for vertebroplasty: Numerical simulations and experimental validation", *Biomechanics and Modeling in Mechanobiology*, vol. 22, no. 4, pp. 1253–1266, 2023.

[56] L. B. Maslov, "Biomechanical model and numerical analysis of tissue regeneration within a porous scaffold", *Mechanics of Solids*, vol. 55, no. 7, pp. 1115–1134, 2020.

[57] P. F. Antonietti, C. Facciolà, A. Russo, and M. Verani, "Discontinuous Galerkin approximation of flows in fractured porous media on polytopic grids", *SIAM Journal on Scientific Computing*, vol. 41, no. 1, pp. A109–A138, 2019.

[58] P. F. Antonietti, C. Facciolà, and M. Verani, "Polytopic discontinuous Galerkin methods for the numerical modelling of flow in porous media with networks of intersecting fractures", *Computers and Mathematics with Applications*, vol. 116, pp. 116–139, 2022.

[59] M. Corti, F. Bonizzoni, and P. F. Antonietti, "Structure preserving polytopal discontinuous Galerkin methods for the numerical modeling of neurodegenerative diseases", *Journal of Scientific Computing*, vol. in press, 2024.

[60] A. Gatti, Z. Hu, T. Smidt, E. G. Ng, and P. Ghysels, "Graph partitioning and sparse matrix ordering using reinforcement learning and Graph Neural Networks", *Journal of Machine Learning Research*, vol. 23, no. 1, p. 13675–13702, 2022.

MOX Technical Reports, last issues

Dipartimento di Matematica
Politecnico di Milano, Via Bonardi 9 - 20133 Milano (Italy)

42/2024 Fois, M.; Katili M. A.; de Falco C.; Larese A.; Formaggia L.
Landslide run-out simulations with depth-averaged models and integration with 3D impact analysis using the Material Point Method

41/2024 Bergonzoli, G.; Rossi, L.; Masci, C.
Ordinal Mixed-Effects Random Forest

40/2024 Carrara, D.; Regazzoni, F.; Pagani, S.
Implicit neural field reconstruction on complex shapes from scattered and noisy data

39/2024 Bartsch, J.; Buchwald, S.; Ciaramella, G.; Volkwein, S.
Reconstruction of unknown nonlienar operators in semilinear elliptic models using optimal inputs

38/2024 Tonini, A., Regazzoni, F., Salvador, M., Dede', L., Scrofani, R., Fusini, L., Cogliati, C., Pontone, G., Vergara, C., Quarteroni, A.
Two new calibration techniques of lumped-parameter mathematical models for the cardiovascular system

Fumagalli, A.; Patacchini, F.S.
Numerical validation of an adaptive model for the determination of nonlinear-flow regions in highly heterogeneous porous media

37/2024 Begu, B.; Panzeri, S.; Arnone, E.; Carey, M.; Sangalli, L.M.
A nonparametric penalized likelihood approach to density estimation of space-time point patterns

36/2024 Torri, V.; Ercolanoni, M.; Bortolan, F.; Leoni, O.; Ieva, F.
A NLP-based semi-automatic identification system for delays in follow-up examinations: an Italian case study on clinical referrals

34/2024 Corti, M.
Exploring tau protein and amyloid-beta propagation: a sensitivity analysis of mathematical models based on biological data

35/2024 Botti, L.; Botti, M.; Di Pietro, D.A.; Massa; F.C.
Stability, convergence, and pressure-robustness of numerical schemes for incompressible flows with hybrid velocity and pressure