Mesh quality agglomeration algorithm for the virtual element method applied to discrete fracture networks

Calcolo (2023) 60:27
https://doi.org/10.1007/s10092-023-00517-5

Mesh quality agglomeration algorithm for the virtual element method applied to discrete fracture networks

Tommaso Sorgente[1] · Fabio Vicini[2,3] · Stefano Berrone[2,3] · Silvia Biasotti[1] · Gianmarco Manzini[1,3] · Michela Spagnuolo[1]

Received: 21 August 2022 / Revised: 13 March 2023 / Accepted: 17 March 2023 /
Published online: 28 April 2023

Abstract
We propose a quality-based optimization strategy to reduce the total number of degrees of freedom associated with a discrete problem defined over a polygonal tessellation with the Virtual Element Method. The presented *Quality Agglomeration* algorithm relies only on the geometrical properties of the problem polygonal mesh, agglomerating groups of neighboring elements. We test this approach in the context of fractured porous media, in which the generation of a global conforming mesh on a Discrete Fracture Network leads to a considerable number of unknowns, due to the presence of highly complex geometries (e.g. thin triangles, large angles, small edges) and the significant size of the computational domains. We show the efficiency and the robustness of our approach, applied independently on each fracture for different network configurations, exploiting the flexibility of the Virtual Element Method in handling general polygonal elements.

Keywords Virtual element method · Fractured media · Mesh regularity · Mesh quality indicators · Optimal rates convergence

Mathematics Subject Classification 65N12 · 65N15

1 Introduction

Over the last fifty years, computer simulations of Partial Differential Equations (PDEs) have dramatically increased their impact on research, design, and production, and are now an indispensable tool for modeling and analyzing phenomena from physics, engineering, biology, and medicine. The most popular techniques for computer-based simulations (e.g., the Finite Element Method) rely on suitable

Fabio Vicini, Stefano Berrone, Silvia Biasotti, Gianmarco Manzini and Michela Spagnuolo have contributed equally to this work.

Extended author information available on the last page of the article

descriptions of geometrical entities, such as the computational domain and its properties, which are generally encoded by a mesh, sometimes also called tessellation. Despite decades of research and significant results, techniques for generating meshes with good geometrical properties are still studied and developed. Current meshing algorithms typically produce an initial mesh, which we expect to be dominated by well-shaped elements, and optionally perform optimization steps to maximize the geometrical quality of the elements. We can broadly group such optimization strategies into two categories [1]: geometrical methods (also called *smoothing* or *untangling*) that keep the mesh connectivity fixed while changing only the locations of the mesh vertices [2–4], and topological methods (also *re-meshing*) that keep the vertices fixed while changing only the connectivity [5, 6]. In both cases, the critical point is the definition of the concept of *quality* for polygonal or polyhedral elements. While there is some concordance in the literature on the notion of quality related to triangular and quadrangular meshes, this concept is still a debated topic when associated with generic polygonal meshes, as proved by the vast number of very different attempts to define a universal quality indicator [7–14]. As a result, it is easier to define mesh optimization strategies that are only related to the optimization of the size of the elements, or the number of incident edges, instead of a generic concept of quality.

In this work, we present an algorithm called *Quality Agglomeration* that automatically optimizes the number of degrees of freedom (DOFs) in the discrete problem defined over a polygonal tessellation. The algorithm acts as a topological optimization method, agglomerating groups of neighboring elements. In particular, it minimizes an energy functional based on the mesh quality indicator introduced in [12], which involves the use of the Virtual Element Method (VEM) [15] in the numerical discretization. The algorithm performs energy minimization via the graph-cut technique (also known as maximum-flow or minimum-cut algorithm [16]), an efficient graph-based technique aimed at splitting a graph into two or more parts, minimizing a certain energy.

We also present an application of our algorithm in the context of fractured porous media. The flow simulation in underground fractured porous media is a fundamental task for a huge number of applications, such as carbon capture processes, the monitoring of aquifers, or the contaminant transport in the subsoil. Among the existing approaches for the simulation of underground phenomena in fractured media [17], the Discrete Fracture Network (DFN) model is particularly suited for scenarios in which the permeability of the surrounding media is negligible with respect to the fracture transmissivities. Indeed, in DFN simulations the porous rock matrix is neglected and the fluid is confined to move on the rock discontinuities, called *fractures*, which act as channels for the flow. The DFN model is characterized by a dimensional reduction of each fracture to a planar polygon immersed in a three-dimensional space, and the coupling conditions are imposed on fracture segment intersections [18]. Due to the deficiency of direct measurements of the subsoil properties, the position, orientation, size, and aperture of each polygonal fracture are generated according to different probability distributions. Furthermore, real fractured media are characterized by a huge computational domain, that involves a large number of unknowns, and often present an

intricate system of intersections at different scales, e.g., large faults are crossed by small fractures. Smart approaches to tackle DFN simulations involve the use of non-conforming meshes at fracture interfaces, combined with constrained optimization problems or high-performance domain decomposition strategies [19–24]. Alternative methods rely on a global conforming mesh on the whole network: this requires a numerical method able to deal with generic polygonal tessellations, as the generation of a standard FEM mesh over a complex domain would be computationally demanding. For this reason, polygonal methods such as the Hybrid High Order [25, 26], the Mimetic Finite Difference [27–29], the Discontinuous Galerkin [30], or the Virtual Element Method (VEM) [31–34] have recently been investigated. This work focuses on the global conforming mesh approach combined with the VEM.

The generation of a polygonal tessellation of a DFN conforming to the fracture interfaces is not a trivial task, due to the large number of fractures typically found in real applications, the different sizes of neighboring domains, and the high complexity of their intersections. In particular, it frequently leads to a high number of degrees of freedom (DOFs) in the discrete problem, and to the generation of numerous elongated elements with aligned small edges. Such "bad-shaped" cells impact on the accuracy of the simulation, producing high condition numbers and numerical errors, while the huge number of DOFs translates into a significant computational cost and, again, an increase of the condition numbers.

For these reasons, we propose the application of the Quality Agglomeration algorithm locally on each fracture of the network, which is able to reduce the total unknowns of the discrete problem with a control on the geometric quality of the elements. To evaluate the effects of the Quality Agglomeration, we run the algorithm over two DFNs of increasing complexity and solve a numerical problem over the original and the agglomerated versions. Then, we compare the results produced by the VEM over the different networks in terms of computational cost, approximation error, and convergence rate.

The paper is organized as follows. We devote Sect. 2 to the description of the mesh quality agglomeration algorithm. In Sect. 3 we introduce the DFN model, the numerical problem to be solved and the VEM discretization used. In Sect. 4 we report the numerical tests performed on two DFNs of increasing complexity. Finally, in Sect. 5 we collect conclusions and possible future research directions.

Throughout the paper we will use the standard definition and notation of Sobolev spaces, norms, and seminorms, cf. [35]; thus, given a bounded, connected subset ω of $\mathbb{R}^2$, we denote the dot product and the norm in $\mathrm{L}^2(\omega)$-space with $(\cdot,\cdot)_\omega$ and $||\cdot||_\omega$ respectively. In addition, $|\cdot|_\omega$ is the seminorm of $\mathrm{H}^1(\omega)$. Given a generic dimensional geometric object $\mathcal{G}$, we indicate by $h_\mathcal{G}$ its diameter, $\boldsymbol{x}_\mathcal{G} = (x_\mathcal{G}, y_\mathcal{G}, z_\mathcal{G})$ its centroid and $|\mathcal{G}|$ its measure (i.e. length or area). We will denote as $\mathbb{P}_n(\mathcal{G})$ the space of polynomials of degree less or equal to $n \in \mathbb{N}$ and we adopt the usual convention $\mathbb{P}_{-1}(\mathcal{G}) = \{0\}$. Finally, similarly to [36], we indicate with $\mathcal{M}_n(\mathcal{G})$ the selected basis for the space $\mathbb{P}_n(\mathcal{G})$ defined as

$$\mathcal{M}_n(\mathcal{G}) := \left\{ m \in \mathbb{P}_n(\mathcal{G}) : m = \left(\frac{\boldsymbol{x} - \boldsymbol{x}_{\mathcal{G}}}{h_{\mathcal{G}}} \right)^{\boldsymbol{s}}, |\boldsymbol{s}| \leq n \right\}. \tag{1}$$

2 Mesh quality agglomeration

Consider the two-dimensional domain Ω, tessellated with an initial polygonal mesh Ω_h. The Quality Agglomeration algorithm aims at reducing the total number of DOFs in the virtual element approximation, keeping control over the mesh geometric quality. By Quality Agglomeration, we mean the process in which we glue together neighbouring elements to form bigger ones. The gluing strategy is driven by the minimization of a functional related to a notion of mesh "quality" suited explicitly for the VEM. Furthermore, we admit a finite number of constraints, so that the gluing process is capable of preserving some selected mesh nodes and edges. In particular, we are interested in operating over the two following types of elements:

1. Elements located in regions where we can assume that the solution does not vary so much, for example, areas of the domain that are distant from geometrical details or features. In such regions, the algorithm can increase the size of the elements without a significant impact on the accuracy of the simulation;
2. "Badly-shaped" or pathological elements according to the chosen notion of quality. The gluing algorithm can often improve the quality of such elements by merging them with neighboring elements.

In what follows, we consider the mesh Ω_h formed by generic polygonal elements E with the boundary ∂E subdivided into edges e. Finally, the positive number h indicates the mesh size, i.e., the finite constant that bounds all the element diameters h_E.

2.1 Mesh quality

The first crucial component of the Quality Agglomeration algorithm is the notion of *mesh quality*. The standard approach in the VEM literature to ensure a good quality mesh is to impose some *geometrical* (or *regularity*) *assumptions* that a mesh must respect to guarantee the optimal behavior of the method. In [12], the authors isolate the four principal assumptions typically required (even if not all simultaneously) for the convergence of the VEM on a given mesh family:

G1: there exists a real number $\rho \in (0, 1)$, independent of h, such that every polygon $E \in \Omega_h$ is star-shaped with respect to a disc with radius $r_E \geq \rho h_E$;
G2: there exists a real number $\rho \in (0, 1)$, independent of h, such that for every polygon $E \in \Omega_h$, each edge $e \in \partial E$ satisfies $|e| \geq \rho h_E$;

G3: there exists a positive integer N, independent of h, such that the number of edges of every polygon $E \in \Omega_h$ is (uniformly) bounded by N;
G4: there exists a positive number δ, independent of h, such that for every polygon $E \in \Omega_h$, the 1-dimensional mesh $\mathcal{I}_E$ representing its boundary can be subdivided into a finite number of disjoint sub-meshes $\mathcal{I}_E^1, \dots, \mathcal{I}_E^N$, and for each $\mathcal{I}_E^i$ it holds that $\max_{e \in \mathcal{I}_E^i} h_e / \min_{e \in \mathcal{I}_E^i} h_e \leq \delta$ (see [37] for a more rigorous definition).

Using these assumptions as absolute conditions that a mesh can only satisfy or violate has been shown to be particularly restrictive [12]. Instead, we can define the quality of a mesh as a measure of *how much* it satisfies the above conditions. This approach is more accurate as it captures small quality differences between meshes and does not exclude a priori all the particular cases of meshes that are only slightly outside the geometrical assumptions.

In [12], the authors derive four scalar functions $\varrho_s : \Omega_h \to [0, 1]$, which measure how a mesh element $E \in \Omega_h$ meets the requirements of assumption **Gs**, for $\mathbf{s} = 1, \dots, 4$. In particular, $\varrho_s = 0$ if E does not respect **Gs**, and the higher ϱ_s the better E is shaped with respect to **Gs**. We briefly report the indicator definitions and refer the reader to [12] and [38] for a more complete discussion:

$$\varrho_1(E) = \frac{|kernel(E)|}{|E|} = \begin{cases} 1 & \text{if } E \text{ is convex} \\ \in (0,1) & \text{if } E \text{ is concave and star-shaped} \\ 0 & \text{if } E \text{ is not star-shaped} \end{cases}$$

$$\varrho_2(E) = \frac{\min(\sqrt{|E|},\ \min_{e \in \partial E} |e|)}{h_E}$$

$$\varrho_3(E) = \frac{3}{\#\{e \in \partial E\}} = \begin{cases} 1 & \text{if } E \text{ is a triangle} \\ \in (0,1) & \text{otherwise} \end{cases}$$

$$\varrho_4(E) = \min_j \frac{\min_{e \in \mathcal{I}_E^j} |e|}{\max_{e \in \mathcal{I}_E^j} |e|}$$

The *kernel* operator computes the kernel of a polygon, intended as the set of points from which the whole polygon is visible, and $\mathcal{I}_E^j$ are all the 1-dimensional disjoint sub-meshes corresponding to the edges of E (we consider each $\mathcal{I}_E^j$ as a mesh as it may contain more than one edge, see [37]) such that $\mathcal{I}_E = \cup_j \mathcal{I}_E^j$, where $\mathcal{I}_E$ is the 1-dimensional mesh induced by ∂E introduced in assumption **G4**.

We combine together the four indicators $\varrho_1, \varrho_2, \varrho_3$ and ϱ_4 into a *quality indicator* $\varrho : \Omega_h \to [0, 1]$, which measures the overall quality of an element $E \in \Omega_h$:

$$\varrho(E) := \sqrt{\frac{\varrho_1(E)\varrho_2(E) + \varrho_1(E)\varrho_3(E) + \varrho_1(E)\varrho_4(E)}{3}}. \tag{2}$$

We have $\varrho(E) \to 1$ if E is a perfectly-shaped element, e.g. an equilateral triangle or a square, $\varrho(E) = 0$ if and only if E is not star-shaped, and $0 < \varrho(E) < 1$ otherwise.

We remark that all indicators $\{\varrho_s\}_{s\in\{1,\dots,4\}}$, and consequently ϱ, only depend on the geometrical properties of the mesh elements; therefore their values can be computed before applying the VEM.

We point out that the local quality indicator (2) is the restriction to a single element of the mesh quality indicator introduced in [12]. A strict correspondence has been proven between the values measured by the mesh quality indicator and the performance of the VEM in terms of approximation errors and convergence rates. In particular, the VEM is likely to produce small approximation errors over meshes with a high quality value in the sense of (2). Moreover, given a collection of refined meshes with decreasing mesh size, the VEM is expected to converge rapidly at the optimal rate if the quality of the meshes remains constant throughout the refinement process (note that ϱ is scale-independent).

2.2 Mesh agglomeration

The second component of the Quality Agglomeration algorithm is the procedure that agglomerates groups of elements in the mesh, following the information provided by the quality indicator. We perform the element agglomeration by solving an optimization problem that balances the number of elements in the mesh and their quality. The energy functional $\mathcal{E} : \Omega_h \to \mathbb{R}$ that we want to minimize is:

$$\mathcal{E} := \sum_{E,E'\in\Omega_h} dc(E,E') + \lambda \sum_{E,E'\in\Omega_h} sc(E,E'). \tag{3}$$

We define the *cost functions* dc and sc from the product space $\Omega_h \times \Omega_h$ to the unit interval [0, 1] as follows:

- the *data cost* (dc) represents the cost of agglomerating two elements $E, E' \in \Omega_h$, and measures the potential quality of the element $E \cup E'$. We define it as:

$$dc(E,E') := \begin{cases} 0 & \text{if } E = E' \\ 1 - \varrho(E \cup E') & \text{if } E \text{ and } E' \text{ are adjacent} \\ 1 & \text{otherwise} \end{cases} \tag{4}$$

  Here, $E \cup E'$ is the boolean union of the two neighboring elements, and ϱ is the elemental quality indicator (2) of the union;
- the *smoothness cost* (sc) encodes information on the structure of the mesh, setting a zero weight to the non-adjacent elements:

$$sc(E,E') := \begin{cases} 1 & \text{if } E \text{ and } E' \text{ are adjacent and } E \neq E' \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

The *agglomeration parameter* $\lambda \in [0, 1]$ balances the relative importance of the two cost functions. By setting $\lambda = 0$ we are essentially saying that all the elements are isolated, and therefore no agglomeration is possible. In the next sections, we use this value to indicate the original input mesh. It would be possible to consider values of

λ greater than 1, but our tests suggest that the most interesting results are obtained within the range [0, 1], see Fig. 2. Empirically, values of λ closer to 1 lead to more aggressive agglomerations.

To minimize the energy functional defined in (3) we interpret the mesh Ω_h as a graph, in which a node represents an element $E \in \Omega_h$. The graph connects two nodes if their corresponding elements are adjacent, i.e., if they share at least one mesh edge. We use the graph-cut technique [39–41] to solve the energy optimization problem, in the implementation provided by the *Multi-label Optimization* method [42]. Graph-cut iterates on the graph nodes, i.e., the mesh elements, and opportunely groups them until the energy associated with the graph undergoes a given threshold. During this process, graph-cut assigns a label to each node to indicate the element(s) that this node should be merged with.

Let L denote the set of all possible labels and $\mathcal{L} : \Omega_h \to L$ the map that assigns a label $l \in L$ to each node $E \in \Omega_h$. In [42] the cost functions are defined in the form $\widetilde{dc} : \Omega_h \times L \to [0, 1]$ and $\widetilde{sc} : L \times L \to [0, 1]$, while we defined both (4) and (5) over $\Omega_h \times \Omega_h$. To align with this notation, we set $L := \{1, \dots, \#\Omega_h\}$ and define the trivial labeling $\widetilde{\mathcal{L}} : \Omega_h \to L$ that bijectively maps each element of the mesh to its *index*, i.e., its position in the array of the mesh data structure containing all the elements. Then, we opportunely compose the cost functions (4) and (5) with the inverse map $\widetilde{\mathcal{L}}^{-1} : L \to \Omega_h$, which, therefore, connects a label $l \in L$ to the (unique) element $E \in \Omega_h$ with index l:

$$\widetilde{dc}(E, l) := dc(E, \widetilde{\mathcal{L}}^{-1}(l)) = dc(E, E'),$$
$$\widetilde{sc}(l_1, l_2) := sc(\widetilde{\mathcal{L}}^{-1}(l_1), \widetilde{\mathcal{L}}^{-1}(l_2)) = sc(E_1, E_2),$$

where E', E_1, and E_2 are the graph nodes whose corresponding elements have indices l, l_1, and l_2, respectively. In other words, the operator $dc(E, E')$ expresses the cost of merging elements E and E', while the operator $\widetilde{dc}(E, l)$ expresses the cost of merging E with the l-th element of the mesh. If the l-th element is E', the trivial labeling will give $\widetilde{\mathcal{L}}^{-1}(l) = E'$ and the two operators coincide.

Substituting $\widetilde{dc}$ and $\widetilde{sc}$ into (3), we obtain a new energy functional, which depends on the particular labeling:

$$\widetilde{\mathcal{E}}(\mathcal{L}) := \sum_{E \in \Omega_h} \widetilde{dc}(E, l_E) + \lambda \sum_{E, E' \in \Omega_h} \widetilde{sc}(l_E, l_{E'}). \tag{6}$$

The minimization problem now reads as $\min_{\mathcal{L} \in \mathcal{P}} \widetilde{\mathcal{E}}(\mathcal{L})$, where $\mathcal{P}$ is the set of all the possible labelings. We solve this problem by using the *alpha-beta swap* algorithm [39]. This technique iteratively segments the graph nodes labeled with a given label α to those with another label β. These two labels change after each iteration, scouting all the possible combinations. Other algorithms exist in the literature, for example, the so-called *alpha-expansion* algorithm, which requires the function sc to be a metric (i.e., to respect the triangular inequality). The results obtained with this latter algorithm are less useful in our context because it generally leads to uneven label distributions. Indeed, the alpha-expansion algorithm tries to expand each label as

much as possible, thus assigning the same label to large part of the domain and different labels only to small isolated areas. This strategy leads to meshes with unbalanced elements.

2.3 The algorithm

We now detail the Quality Agglomeration algorithm for the VEM. Given an input mesh $\widetilde{\Omega}_h$, we build the corresponding graph by generating a node for each element in the mesh and connecting the nodes corresponding to adjacent elements. In the case of a constrained mesh edge that has to be preserved, we consider the elements sharing that edge as non-adjacent. We initialize the node labels with the trivial labeling $\widetilde{\mathcal{L}}$, see Fig. 1a. Then, we run the graph-cut technique with a selected value of the parameter λ, and find new labels for the elements (Fig. 1b). Note that graph-cut only operates over the labels without actually modifying the mesh. Graph-cut reaches convergence when the energy term (6) attains a (local) minimum. This task is typically accomplished in a few iterations. Last, a post-processing step is required to merge all elements sharing the same label (Fig. 1c) and create the agglomerated mesh Ω_h. To further improve the mesh quality, the algorithm merges aligned edges in the newly generated elements (Fig. 1d), while preserving possible constraints on nodes or edges of the initial mesh $\widetilde{\Omega}_h$.

Since the agglomerated mesh Ω_h is the solution to a minimization problem, it is also optimal in the number of elements. However, we can compute different optimal versions of the same mesh $\widetilde{\Omega}_h$ by choosing different values of the parameter λ in (6). Fig. 2 gives a hint on how to choose the value of λ: the difference between $\lambda = 0.23$ and $\lambda = 0.25$ is significant, while from $\lambda = 0.25$ to $\lambda = 1$ the output does not change much. Such effects highly depend on the geometric properties of the considered mesh; however, we noted that for $\lambda > 1$ we hardly see differences on the agglomerated meshes. Higher values of λ provide agglomerated meshes containing a smaller number of elements, edges, nodes, and, therefore, degrees

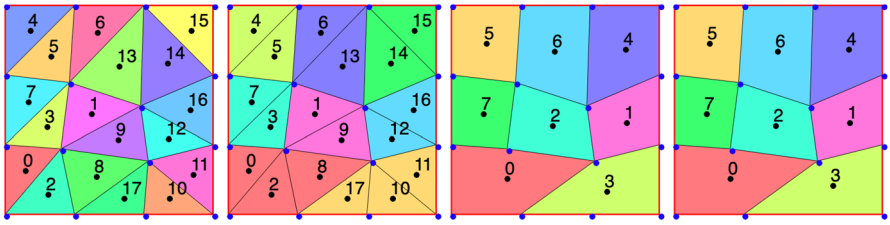


Fig. 1 Visualization of the agglomeration algorithm, with elements colored with respect to their label and elements' indices reported in black. **a** Initial mesh $\widetilde{\Omega}_h$: every element has a different label, corresponding to its index; **b** after the graph-cut algorithm, some elements share the same label (color), while maintaining distinct indices; **c** agglomerated mesh Ω_h: new elements are created with new indices and new labels; **d** merging of aligned edges: the vertex at the bottom of element 3 is removed

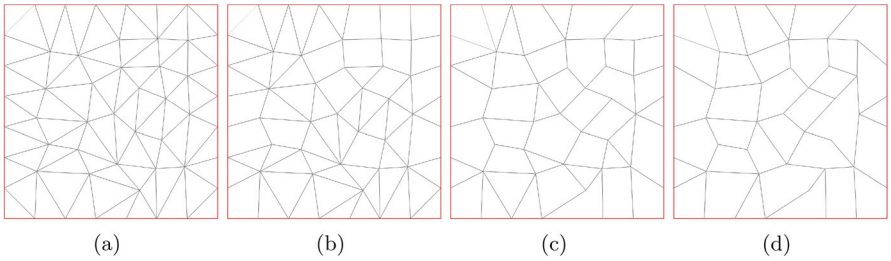


Fig. 2 Impact of the paramenter λ on the allgomeration algorithm. **a** Original mesh ($\lambda = 0$) with 54 vertices, 130 edges, and 78 elements; **b** agglomeration with $\lambda = 0.23$: 0 vertices, 16 edges, and 16 elements removed; **c** $\lambda = 0.25$: 4 vertices, 40 edges, and 36 elements removed; **d** $\lambda = 1.0$: 5 vertices, 49 edges, and 44 elements removed

of freedom of the VEM space, thus requiring a smaller computational cost for the VEM. At the same time, we expect the errors produced by the agglomerated meshes to be slightly higher, as for every removed DOF we have less information on the numerical solution. We are interested in understanding how this reduction of degrees of freedom impacts the accuracy of the VEM, and if it affects the convergence rate of the method.

3 DFN problem formulation

To test the Quality Agglomeration algorithm, we introduce the steady-state flow problem in DFNs as a possible application. We consider a network domain $\Omega \subset \mathbb{R}^3$ made of a finite number $\bar{i}$ of fractures F_i, $i \in \mathcal{I} := \{1, \dots, \bar{i}\}$; thus, $\Omega := \bigcup_{i \in \mathcal{I}} F_i$, such that each F_i has at least one intersection with at least one F_j, for $j \in \mathcal{I} \setminus \{i\}$. In the DFN model, each F_i is represented by a two-dimensional polygonal tessellation oriented in $\mathbb{R}^3$ that approximates a geological fracture immersed in the impervious rock matrix, as done in [18, 33, 34, 43, 44]. With this assumption, the network results in a collection of two-dimensional elements. Therefore, we refer to it with the same symbol Ω of Sect. 2 even if it is not properly contained in $\mathbb{R}^2$.

Let $\partial\Omega := \bigcup_{i \in \mathcal{I}} \partial F_i$ be the domain boundary. We split it into a Dirichlet part Γ_D and a Neumann part Γ_N, such that $\partial\Omega = \Gamma_D \cup \Gamma_N$, $\Gamma_D \cap \Gamma_N = \emptyset$, and $|\Gamma_D| \neq 0$. The fractures intersect along a finite number $\bar{m}$ of segments, denoted as S_m, with $m \in \mathcal{M} := \{1, \dots, \bar{m}\}$. For the sake of simplicity, we assume that each intersection is formed by exactly two fractures, i.e., $S_m := F_i \cap F_j$, fixing a unique pair of fracture indices $\sigma_m = \{i, j\}$ for each $m \in \mathcal{M}$. We point out that this is a simplifying assumption that does not alter the results. Finally, we denote by $\mathcal{M}_i$ the set of the indices of the fracture intersections S_m which lie on F_i, i.e., $\mathcal{M}_i := \{m \in \mathcal{M} : S_m \cap F_i \neq \emptyset\}$. In what follows, ∇_i represents the tangential component of the tridimensional gradient operator ∇ on the plane of F_i and the subscript i on a quantity indicates its restriction to F_i.

We seek the distribution of the hydraulic head u in the whole network Ω; we prescribe the Dirichlet boundary condition on Γ_D through function $g \in \mathrm{H}^{\frac{1}{2}}(\Gamma_D)$ and set the homogeneous Neumann boundary condition on Γ_N. We define the functional spaces:

$$V^D := \{v : v_{|\Gamma_D} = g,\ v_i \in \mathrm{H}^1(F_i)\ \forall i \in \mathcal{I},\ v_{i|S_m} = v_{j|S_m}\ \forall m \in \mathcal{M}\},$$
$$V := \{v : v_i \in \mathrm{H}^1_0(F_i)\ \forall i \in \mathcal{I},\ v_{i|S_m} = v_{j|S_m}\ \forall m \in \mathcal{M}\}.$$

The weak formulation reads: find $u \in V^D$ such that $u - u^D \in V$ with $u^D \in V^D$ and, for all $v \in V$,

$$\sum_{i\in\mathcal{I}} \left(\mathbb{K}_i \nabla_i u_i, \nabla_i v_i\right)_{F_i} = \sum_{i\in\mathcal{I}} \left(f_i, v_i\right)_{F_i}, \tag{7}$$

where $\mathbb{K}_i : \mathbb{R}^2 \to \mathbb{R}^{2\times 2}$ is the in-plane transmissivity on the fracture. According to [45, Theorem 2.7.7], problem (7) is well posed since the symmetric bilinear form

$$a(v, w) := \sum_{i\in\mathcal{I}} a^{F_i}(v, w) = \sum_{i\in\mathcal{I}} \left(\mathbb{K}_i \nabla_i w_i, \nabla_i v_i\right)_{F_i} \tag{8}$$

is coercive and continuous on V.

3.1 Network discretization

We describe the approach used to construct an optimal polygonal tessellation Ω_h on Ω, globally conforming at the fracture intersections, exploiting the Quality Agglomeration algorithm of Sect. 2.3. The approach is composed of three steps; we show an illustrative example on Fig. 3.

First, we independently discretize each fracture domain $F_i \in \Omega$ by a classical triangular planar mesh $\mathcal{T}_h^i$ of given size h. On each $\mathcal{T}_h^i$, given the set of local fracture segments S_m with $m \in \mathcal{M}_i$, we split all the cells $T \in \mathcal{T}_h^i$ that intersect the segments S_m by the direction of the segments [33]. Thus, on each F_i we obtain a local polygonal mesh $\mathcal{P}_h^i$, which is conforming to all S_m with $m \in \mathcal{M}_i$, see Fig. 3a. Since we do not implement any particular geometrical smoothing technique on this cutting

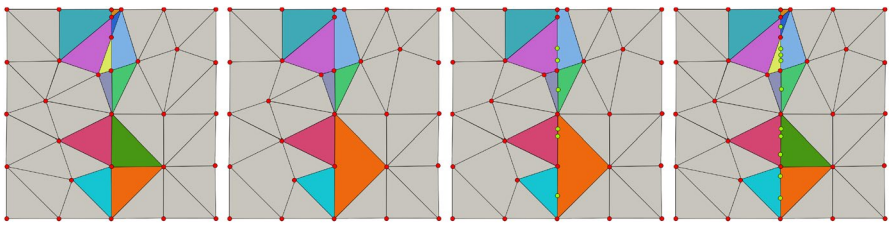


Fig. 3 Visualization of a single fracture mesh during the discretization process, with elements around the intersection colored. **a** Local conforming polygonal mesh $\mathcal{P}_h^i$ on fracture F_i: the original triangular elements around the intersection segment are split in sub-polygons; **b** locally optimal quality discretization Ω_h^i with new agglomerated polygons; **c** global conforming mesh Ω_h with the new polygon vertices (green points) added along the intersection segment; **d** global conforming mesh Ω_h like it would be without the agglomeration process

phase, the meshes $\mathcal{P}_h^i$ may present small edges and elongated "bad-shaped" cells, in the sense of the geometrical assumptions from Sect. 2.1.

Second, the Quality Agglomeration algorithm described in Sect. 2.3 processes all the meshes $\mathcal{P}_h^i$ independently and imposes the constraints on the mesh vertices that are the segment endpoints of each fracture intersections S_m, $m \in \mathcal{M}_i$. Thus, on each domain F_i we obtain a locally optimal quality discretization Ω_h^i that still respects the conformity constraints along boundaries and interfaces S_m with $m \in \mathcal{M}_i$, see Fig. 3b.

Finally, we collect the optimal meshes Ω_h^i to generate the global polygonal conforming mesh Ω_h on the whole network. The conforming operation only works on the intersection segments S_m: for each $m \in \mathcal{M}$, with $\sigma_m = \{i, j\}$, we unify the two sets of mesh nodes in $\mathbb{R}^3$ of Ω_h^i and Ω_h^j lying on the segment S_m, and create new mesh elements $E \in \Omega_h$ that have aligned edges in correspondence of the new nodes, see Fig. 3c. We point out that, at the end of the agglomeration process, aligned points lying on S_m are removed (e.g., the vertex shared by the purple, yellow, light blue and blue elements in Fig. 3a). Therefore, the number of new points to be added during the global conforming step gets considerably smaller. With this approach, we obtain that each element $E \in \Omega_h$ lies in only one $F_i \in \Omega$. Therefore, quantities introduced locally on E are considered with respect to the two-dimensional tangential reference system of F_i, and we omit in what follows the suffix $_i$ on all the operators (e.g., ∇_i becomes ∇).

In Fig. 3d we present the result of the global conforming operation without the agglomeration step. The difference with Fig. 3c is significant, both in terms of number of degrees of freedom and of geometrical quality of the elements. Moreover, if we wanted to apply a standard FEM on this mesh, we should connect each green vertex to a red one to obtain a triangular mesh. This visual example gives a first hint on the impact of our technique, better analyzed in Sect. 4.

3.2 Virtual element approximation

We remark that $\mathbb{P}_n(E)$ is built on the reference system that is tangential to the fracture F_i to which the mesh element E belongs. We define $\Pi^{\nabla}_{n,E} : \mathrm{H}^1(E) \to \mathbb{P}_n(E)$ as the $\mathrm{H}^1(E)$-orthogonal projection operator, computed for any $p \in \mathbb{P}_n(E)$ and $v \in \mathrm{H}^1(E)$ with the conditions:

$$\begin{cases} \left(\nabla \Pi^{\nabla}_{n,E} v, \nabla p\right)_E = (\nabla v, \nabla p)_E \\ \left(\Pi^{\nabla}_{n,E} v, 1\right)_{\partial E} = (v, 1)_{\partial E} & \text{if } n = 1 \\ \left(\Pi^{\nabla}_{n,E} v, 1\right)_E = (v, 1)_E & \text{otherwise} \end{cases}$$

Similarly, we let $\Pi^0_{n,E}$ and $\Pi^0_{n,E}\nabla$ denote the $\mathrm{L}^2(E)$-orthogonal projection operators on $\mathbb{P}_n(E)$ for functions v in $\mathrm{H}^1(E)$ and on $[\mathbb{P}_n(E)]^2$, respectively.

We locally define the Virtual Element space of order $k \geq 1$ on E as

$$V_h^{k,E} := \Big\{ v \in \mathrm{H}^1(E) : \Delta v \in \mathbb{P}_k(E),\ v_{|\partial E} \in C^0(\partial E),\ v_{|e} \in \mathbb{P}_k(e)\ \forall e \in E,$$
$$(v,p)_E = \Big(\Pi_{k,E}^{\nabla} v, p\Big)_E\ \forall p \in \mathbb{P}_k(E) \setminus \mathbb{P}_{k-2}(E)\Big\}.$$

Given a function $v \in V_h^{k,E}$, we select the following standard degrees of freedom on the element E:

- the values of v in the vertices of E;
- if $k > 1$, the values of v on the $k-1$ internal Gauss-Lobatto quadrature points on every edge $e \in E$;
- if $k > 1$, the $\frac{k(k-2)}{2}$ moments of v on E: $\frac{1}{|E|}\big(v, m_j\big)_E$, $\forall m_j \in \mathcal{M}_{k-2}(E)$.

The chosen DOFs are unisolvent for $V_h^{k,E}$, and all the projection operators $\Pi_{k,E}^{\nabla}$, $\Pi_{k,E}^0$, and $\Pi_{k-1,E}^0 \nabla$ are computable [15]. Finally, using the Lagrangian basis functions with respect to the DOFs as a basis for $V_h^{k,E}$, we define the global discrete space V_h^k on the whole domain as

$$V_h^k := \Big\{ v \in V : v_{|E} \in V_h^{k,E}\ \forall E \in \Omega_h \Big\}.$$

We can now define the discrete counterpart of the bilinear form introduced in (8) as $a_h : V_h^k \times V_h^k \to \mathbb{R}$, such that for all $v, w \in V_h^k$:

$$a_h(v,w) := \sum_{E \in \Omega_h} a_h^E(v,w),$$

with

$$a_h^E(v,w) := \Big(\mathbb{K}_i \Pi_{k-1,E}^0 \nabla w_{|F_i}, \Pi_{k-1,E}^0 \nabla v_i\Big)_{F_i} + S_E^{F_i}(v - \Pi_{k,E}^{\nabla} v, w - \Pi_{k,E}^{\nabla} w),$$

where F_i is the fracture to which the mesh element E belongs.

The bilinear form $S_E^{F_i}$ is selected such that

$$\exists\, \alpha, \beta > 0 :\ \alpha\, a_E^{F_i}(u,u) \le a_h^E(u,u) \le \beta\, a_E^{F_i}(u,u)\ \forall u \in V_h^{k,E},$$

where $a_E^{F_i}$ is the restriction to the mesh element E of the bilinear form a^{F_i} introduced in (8). Different choices for $S_E^{F_i}$ are used in the literature, see for example [46]. In this work we select the typical form

$$S_E^{F_i}(v,w) := \big|\big|\mathbb{K}_i\big|\big|_{\mathrm{L}^\infty(E)} \sum_{\ell=1}^{\#V_h^{k,E}} \mathrm{dof}_\ell^E(v)\, \mathrm{dof}_\ell^E(w), \tag{9}$$

where $\mathrm{dof}_\ell^E(\cdot)$ is the operator that selects the ℓ-th degree of freedom of $V_h^{k,E}$.

The discrete formulation of problem (7) reads as: Find $u_h \in V_h^k$ such that, for all $v_h \in V_h^k$, it holds

$$a_h(u_h, v_h) = \sum_{E \in \Omega_h} \left(f, \Pi^0_{k-1,E} v_h \right)_E. \tag{10}$$

For a detailed discussion about the approximation of the source term we refer to [15]. Finally, we can prove the well-posedness of problem (10) and the optimal convergence rates measured in the numerical results of Sect. 4 by using standard arguments for Virtual Element methods, see [47].

4 Numerical results

We present some numerical results for two increasingly complex networks: the first deals with a known hydraulic head in a simple setting, whereas the second simulates a quite realistic DFN configuration.

In both cases, we apply the Quality Agglomeration algorithm proposed in Sect. 2 on each F_i with $i \in \mathcal{I}$, using the same agglomeration parameter for all the fractures of the network Ω. We associate the value $\lambda = 0.0$ to the non-agglomerated strategy, and compare two different λ values, namely 0.25 and 1.0, to test a moderate and an aggressive agglomeration strategy, respectively. We stress that the mesh optimization approach is performed independently on each domain F_i. Thus, we can tackle large networks coming from real applications with a naturally distributed parallel strategy.

We compare the values of the energy functional (6) measured in the global polygonal conforming mesh Ω_h of the network before and after the optimization process. We test the effectiveness of the algorithm by comparing the performance of the VEM from Sect. 3.2 with $k = \{1, 2, 3\}$ for each λ value. We set a constant transmissivity $\mathbb{K}_i = \mathbb{I}$ for all $i \in \mathcal{I}$, and solve the linear system generated by problem (10) directly, exploiting the Cholesky factorization of the C++ Eigen library, see [48].

4.1 Network 1: simple DFN with known solution

In the first test, we analyze the convergence properties of the proposed method on a simple network that is composed by three fractures aligned with the reference system:

$$\begin{aligned}
F_1 &= \{(x, y, z) \in \mathbb{R}^3 : -1.0 \le x \le 0.5,\ -1.0 \le y \le 1.0,\ z = 0.0\},\\
F_2 &= \{(x, y, z) \in \mathbb{R}^3 : -1.0 \le x \le 0.0,\ y = 0.0,\ -1.0 \le z \le 1.0\},\\
F_3 &= \{(x, y, z) \in \mathbb{R}^3 : x = 0.5,\ -1.0 \le y \le 1.0,\ -1.0 \le z \le 1.0\}.
\end{aligned}$$

These fractures intersect along three interfaces, S_1, S_2, and S_3, see Fig. 4.

We test three tessellations of decreasing size, labeled M1, M2, and M3. We first discretize the network with three triangular meshes fixing the maximum area of the triangular elements to 10^{-1}, 10^{-2}, and 10^{-3}. Then, we cut the elements along the fracture intersection segments as described in Sect. 3.1. Finally, we apply the Quality

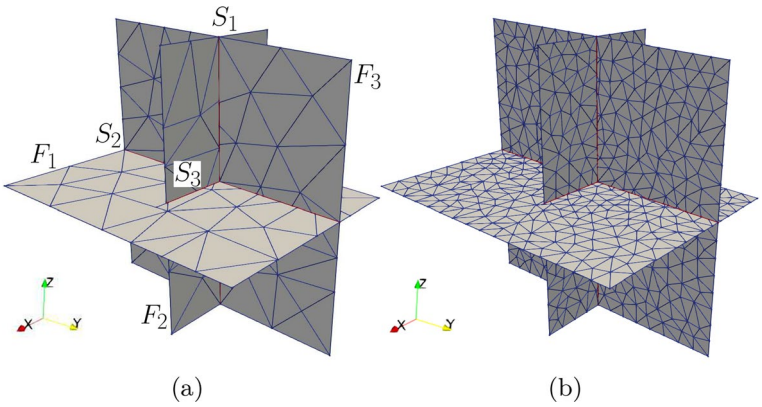


Fig. 4 Network 1; discretization with mesh M1 **a** and M2 **b**, with constrained edges along the interfaces marked in red

Agglomeration algorithm described in Sect. 2 to M1, M2, and M3, with $\lambda = 0.25$ and $\lambda = 1.0$. In Fig. 5a we present a visualization of the effects of the agglomeration: the algorithm deletes several edges from the original mesh with $\lambda = 0.0$ (in blue). By observing the localization of the quality indicator ϱ on a single fracture (Fig. 5b), we notice how the most pathological elements (in red) are around the interfaces.

Table 1 summarizes the results of the optimization process applied to meshes M1, M2, and M3 on fracture F_3 (we do not report F_1 and F_2, as the measured values were comparable). We analyze the energy functional (3) and compare its value $\mathcal{E}_1$ over the original meshes against its value $\mathcal{E}_2$ over the meshes agglomerated with the two λ values. In columns $(\mathcal{E}_1 - \mathcal{E}_2)$ we report the percentage of energy saved by the optimization process, compared to the total original energy of the non-agglomerated mesh. In columns $\mathcal{E}_{2,\text{dc}}$ and $\mathcal{E}_{2,\text{sc}}$, we report the contribution of the data cost and the smoothness cost to the total energy $\mathcal{E}_2$. We remark that, while $\mathcal{E}_{1,\text{dc}}$ and $\mathcal{E}_{1,\text{sc}}$ are equal over the same mesh, their combination $\mathcal{E}_1$ also depends on the value of λ. We can observe how the process always leads to a reduction of the total energy within a small number of iterations (this latter is reported in column *It*). However, using $\lambda = 0.25$ leads to a more conservative strategy (around 2% of energy saving), counter to the aggressive approach of using $\lambda = 1.0$ (more than 30% of energy saving).

In Table 2, we report the number of elements $|\Omega_h|$ of the meshes for the different λ values, and the relative number of DOFs (which varies according to the VEM order k). We observe that $|\Omega_h|$ is roughly 30% smaller for $\lambda = 0.25$ and 70% smaller for $\lambda = 1.0$ than the value for the original mesh for $\lambda = 0$. As far as DOFs are concerned, we recall from Sect. 3.2 that, for $k = 1$, the degrees of freedom correspond to the mesh vertices; thus, since the agglomeration algorithm mainly removes edges and elements, the number of DOFs remains almost untouched in the linear formulation, especially with $\lambda = 0.25$. As soon as we raise the order, however, the difference becomes significant: around 50% reduction in the case of $k = 2$ and $\lambda = 1.0$, and slightly more for $k = 3$.

We set the numerical problem by imposing the Dirichlet boundary conditions in accordance with the exact solution

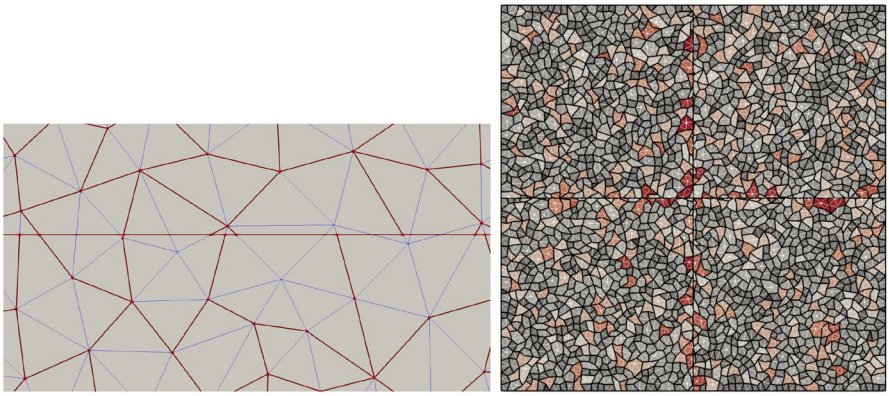


Fig. 5 Network 1, mesh M3; **a** visual comparison between fracture F_3 with $\lambda = 0.0$ (blue lines) and $\lambda = 1.0$ (red lines) **b** fracture F_3 with $\lambda = 1.0$ colored w.r.t. the ϱ value on each cell, from red ($\varrho \approx 0$) to white ($\varrho \approx 1$)

Table 1 Network 1; energy functional over F_3 before ($\mathcal{E}_1$) and after ($\mathcal{E}_2$) the optimization, with the detail of the data cost ($\mathcal{E}_{dc}$) and smoothness cost ($\mathcal{E}_{sc}$) contribution

	λ	*It*	ε_1	$(\varepsilon_1 - \varepsilon_2)\%$	$\varepsilon_{2,\mathrm{dc}}\,\%$	$\varepsilon_{2,\mathrm{sc}}\,\%$
M1	0.25	4	2.12×10^3	1.23	19.81	80.19
M1	1.00	5	8.38×10^3	37.87	37.85	62.15
M2	0.25	4	1.62×10^5	1.95	26.72	73.28
M2	1.00	6	6.47×10^5	33.36	28.44	71.56
M3	0.25	5	1.50×10^7	2.29	28.76	71.24
M3	1.00	9	5.98×10^7	32.38	23.11	76.89

The percentage in the fourth column is computed with respect to $\mathcal{E}_1$, and in the last two columns with respect to $\mathcal{E}_2$. Column *It* shows the number of iterations needed to converge

$$u_1(x,y) = -\frac{1}{10}\left(\frac{1}{2}+x\right)[x^3 + 8xy(x^2+y^2)\mathrm{atan2}(y,x)] \text{on } F_1,$$
$$u_2(x,z) = -\frac{1}{10}\left(\frac{1}{2}+x\right)x^3 + \pi\frac{4}{5}\left(\frac{1}{2}+x\right)x^3|z| \text{on } F_2,$$
$$u_3(y,z) = y(y-1)(y+1)z(z-1) \text{on } F_3.$$

In Fig. 6 we show the solutions computed with the proposed method with $k = 3$ on the mesh M2 for the three different levels of agglomeration. We can appreciate that no qualitative differences can be observed in the three results, despite a significant reduction of the DOFs (up to 50%) in the agglomeration process, see Table 2c.

In the middle columns of Table 2 we show the global approximation errors

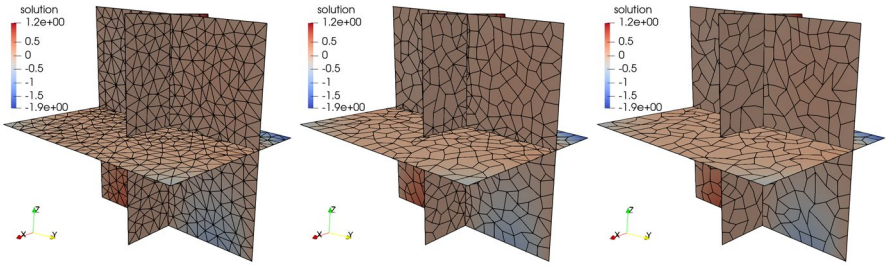
$$||e_h||_{\mathrm{L}^2} := ||u - u_h||_{\mathrm{L}^2(\Omega)}, \qquad ||e_h||_{\mathrm{H}^1} := |u - u_h|_{\mathrm{H}^1(\Omega)}.$$

Table 2 Network 1; analysis of the numerical errors for the different meshes and VEM orders k

	λ	$\lvert\Omega_h\rvert$	DOFs	$\lvert\lvert e_h\rvert\rvert_{L^2}$	$\lvert\lvert e_h\rvert\rvert_{H^1}$	NNZ %	cond ($\mathbb{A}$)
(a) $k = 1$							
M1	0.00	184	131	1.7758×10^{-1}	1.6816×10^{0}	11.87	1.39×10^{2}
M1	0.25	143	131	1.9084×10^{-1}	1.6207×10^{0}	12.77	1.10×10^{2}
M1	1.00	52	92	2.8219×10^{-1}	1.9004×10^{0}	23.44	6.87×10^{1}
M2	0.00	1571	947	1.6651×10^{-2}	5.4935×10^{-1}	1.12	1.29×10^{3}
M2	0.25	972	947	2.3599×10^{-2}	6.0055×10^{-1}	1.30	1.27×10^{3}
M2	1.00	484	788	3.8770×10^{-2}	7.1998×10^{-1}	2.01	6.94×10^{2}
M3	0.00	14,548	7,865	1.6124×10^{-3}	$1.7183.10^{-1}$	0.11	1.69×10^{4}
M3	0.25	8,112	7,865	2.1136×10^{-3}	$1.7904.10^{-1}$	0.13	1.68×10^{4}
M3	1.00	4,687	7,065	3.8983×10^{-3}	$2.2326.10^{-1}$	0.18	9.39×10^{3}
(b) k = 2							
M1	0.00	184	629	1.2251×10^{-2}	2.6309×10^{-1}	2.68	3.41×10^{3}
M1	0.25	143	547	1.6465×10^{-2}	3.1200×10^{-1}	3.62	3.14×10^{3}
M1	1.00	52	287	4.4063×10^{-2}	5.5645×10^{-1}	10.72	1.42×10^{3}
M2	0.00	1,571	5,035	3.6072×10^{-4}	2.6098×10^{-2}	0.29	3.83×10^{4}
M2	0.25	972	3,837	8.9759×10^{-4}	4.4846×10^{-2}	0.49	3.21×10^{4}
M2	1.00	484	2,543	1.7383×10^{-3}	6.8635×10^{-2}	1.01	2.05×10^{4}
M3	0.00	14,548	44,825	1.1458×10^{-5}	2.5864×10^{-3}	0.03	1.20×10^{6}
M3	0.25	8,112	31,953	2.4618×10^{-5}	4.0638×10^{-3}	0.05	8.84×10^{5}
M3	1.00	4,687	23,503	5.4295×10^{-5}	6.6139×10^{-3}	0.09	6.28×10^{5}
(c) k = 3							
M1	0.00	184	1,311	7.2477×10^{-4}	2.2584×10^{-2}	1.85	9.03×10^{6}
M1	0.25	143	1,106	1.8690×10^{-3}	3.7121×10^{-2}	2.53	8.30×10^{6}
M1	1.00	52	534	1.1745×10^{-2}	1.2751×10^{-1}	7.97	4.07×10^{6}
M2	0.00	1,571	10,694	6.6740×10^{-6}	6.7445×10^{-4}	0.20	3.56×10^{8}
M2	0.25	972	7,699	3.4226×10^{-5}	1.8136×10^{-3}	0.35	2.98×10^{8}
M2	1.00	484	4,782	8.4998×10^{-5}	3.6409×10^{-3}	0.77	1.94×10^{8}
M3	0.00	14,548	96,333	6.9926×10^{-8}	2.1951×10^{-5}	0.02	2.04×10^{11}
M3	0.25	8,112	64,153	3.1432×10^{-7}	5.3694×10^{-5}	0.04	1.50×10^{11}
M3	1.00	4,687	44,628	8.6702×10^{-7}	1.1447×10^{-4}	0.07	1.07×10^{11}

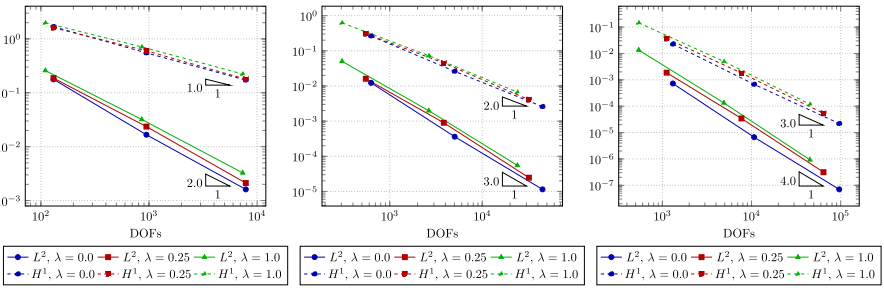
λ is the agglomeration parameter, $\lvert\Omega_h\rvert$ the number of elements in the mesh, e_h the discrete error, $\mathbb{A}$ the stiffness matrix and NNZ its non-zero elements

Fig. 7 shows the error curves versus the total number of degrees of freedom. The slopes of these curves reflect the convergence rate α for the polynomial degrees $k = 1, 2, 3$. Optimal convergence rates are evident in all plots, and the rate lines are similar in all the agglomerations. These optimal rates are achieved thanks to the global conformity of the mesh, despite the low regularity of the solution around the



(a) $\lambda = 0.0$ (b) $\lambda = 0.25$ (c) $\lambda = 1.0$

Fig. 6 Network 1, mesh M2; solution computed with $k = 3$ for each λ value

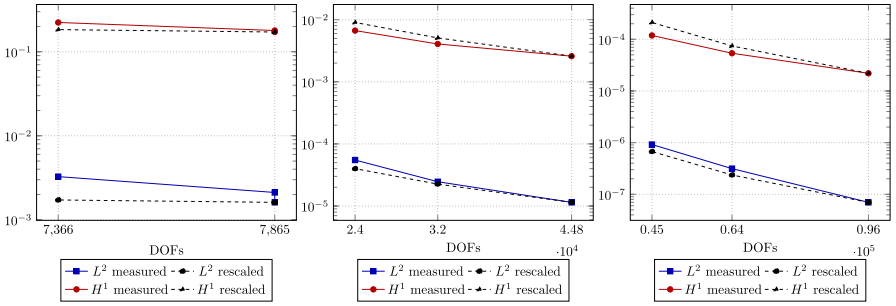


(a) $k = 1$ (b) $k = 2$ (c) $k = 3$

Fig. 7 Network 1; convergence curves of L^2 and H^1 errors for different k values

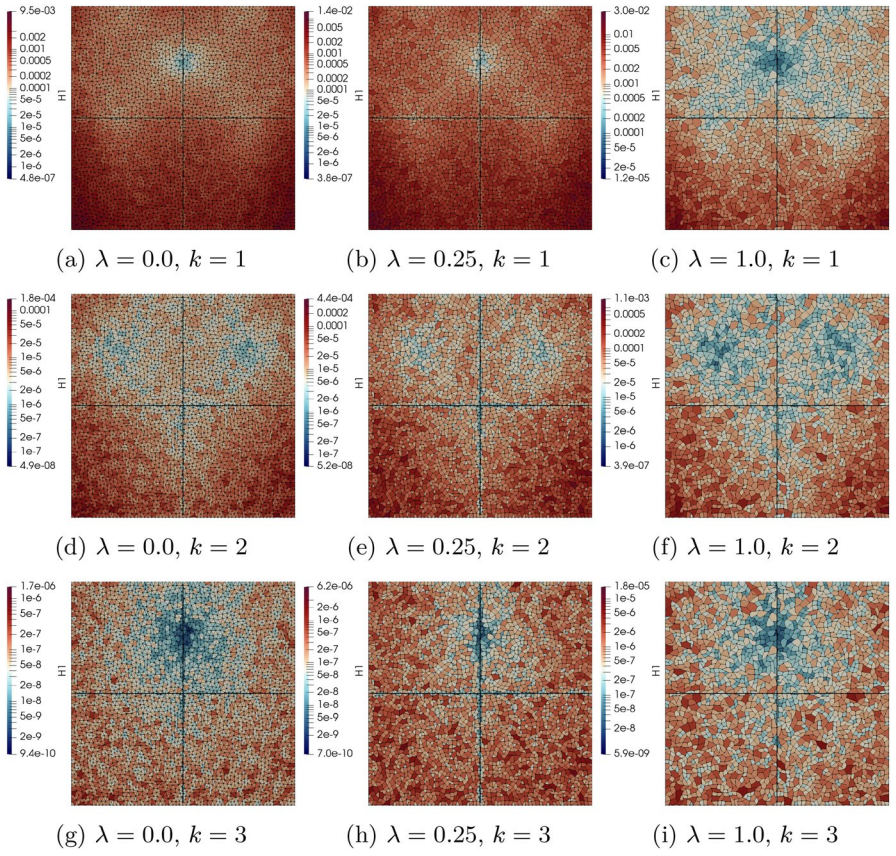
extremity of S_3 that falls inside F_1, see Fig. 4. Note that the dots relative to agglomerated meshes are shifted leftwards, as they contain smaller numbers of DOFs. To analyze pointwise the errors produced by an agglomerated mesh, we rescale the errors measured in the original mesh by the number of DOFs of the agglomerated mesh. Let $\left|\left|e_{h_1}\right|\right|_\star$ and $\left|\left|e_{h_2}\right|\right|_\star$ for $\star \in \left\{\mathrm{L}^2, \mathrm{H}^1\right\}$ the errors measured on two meshes with different mesh size h_1 and h_2. We expect $\left|\left|e_{h_1}\right|\right|_\star h_2^\alpha \approx \left|\left|e_{h_2}\right|\right|_\star h_1^\alpha$. Thus, in the plots of Fig. 8, we compare the errors $||e_h||_{\mathrm{L}^2}$ and $||e_h||_{\mathrm{H}^1}$ and the expected errors obtained by rescaling the non-optimized error norms $||e_h||_{\mathrm{L}^2}^{\lambda=0}$ and $||e_h||_{\mathrm{H}^1}^{\lambda=0}$ with the corresponding convergence rates; the order of magnitude of the two norms are comparable even when the reduction of DOFs is greater than 50%, see Table 2b as a reference.

In particular, when the order of the VEM space increases (Fig. 8b, c) the measured errors in the H^1-seminorm become slightly lower than the expected ones. This fact seems corroborated by Fig. 9, in which we report the localization of $||e_h||_{\mathrm{H}^1}$ on the domain F_3 for the finer mesh M3. Indeed, we can note from these images that the error roughly remains in the same order of magnitude across the agglomeration process (see the color bar at the left of each subplot), and this is more evident for the high VEM orders. Moreover, the blue zones, which correspond to areas with lower



(a) $k = 1$ (b) $k = 2$ (c) $k = 3$

Fig. 8 Network 1, mesh M3; measured errors vs expected errors without mesh optimization



(a) $\lambda = 0.0,\ k = 1$ (b) $\lambda = 0.25,\ k = 1$ (c) $\lambda = 1.0,\ k = 1$

(d) $\lambda = 0.0,\ k = 2$ (e) $\lambda = 0.25,\ k = 2$ (f) $\lambda = 1.0,\ k = 2$

(g) $\lambda = 0.0,\ k = 3$ (h) $\lambda = 0.25,\ k = 3$ (i) $\lambda = 1.0,\ k = 3$

Fig. 9 Network 1, mesh M3; localization of the error $||e_h||_{H^1}$ on each cell of F_3

error values, seem to have a greater extension in the optimized meshes, meaning that the error is more evenly distributed in those cases.

Table 3 Network 1; maximum discrepancy of the projection matrices $\Pi^{\nabla}_{k,*}$ and $\Pi^{0}_{k,*}$ measured in the mesh elements; note that for $k < 3$ we have $\Pi^{\nabla}_{k,*} = \Pi^{0}_{k,*}$

	λ	$\lvert\Pi^{\nabla}_{1,*}\mathcal{D} - \mathcal{I}\rvert_2$	$\lvert\Pi^{\nabla}_{2,*}\mathcal{D} - \mathcal{I}\rvert_2$	$\lvert\Pi^{\nabla}_{3,*}\mathcal{D} - \mathcal{I}\rvert_2$	$\lvert\Pi^{0}_{3,*}\mathcal{D} - \mathcal{I}\rvert_2$
M1	0.00	5.9506×10^{-16}	6.7983×10^{-13}	1.7733×10^{-11}	7.9387×10^{-11}
M1	0.25	6.9427×10^{-16}	6.7983×10^{-13}	1.7735×10^{-11}	7.9387×10^{-11}
M1	1.00	7.3183×10^{-16}	5.7589×10^{-13}	1.5569×10^{-11}	2.0760×10^{-11}
M2	0.00	3.8745×10^{-15}	1.6826×10^{-11}	1.0390×10^{-9}	1.2036×10^{-9}
M2	0.25	3.9257×10^{-15}	1.6826×10^{-11}	1.0387×10^{-9}	1.2032×10^{-9}
M2	1.00	3.3886×10^{-15}	1.3492×10^{-11}	9.0217×10^{-10}	7.5128×10^{-10}
M3	0.00	5.7185×10^{-15}	8.9036×10^{-10}	8.0172×10^{-7}	8.6223×10^{-7}
M3	0.25	5.7185×10^{-15}	8.9036×10^{-10}	8.0482×10^{-7}	8.5470×10^{-7}
M3	1.00	3.0768×10^{-15}	8.9581×10^{-10}	2.4545×10^{-7}	2.6581×10^{-7}

To conclude the analysis, in the last columns of Table 2 we report the number NNZ of non-zero elements and the condition number cond($\mathbb{A}$) of the stiffness matrix associated with the discrete problem (10). From the non-zero values measurements, we can assert that the optimization process does not excessively affect the sparsity pattern of matrix $\mathbb{A}$, since the percentage values for the different λ values are quite close. On the other hand, we observe that the condition number of the stiffness matrix is under control for $k = 1, 2$, and increases for $k = 3$. This growth is typical when using the basis (1) as the polynomial base for the VEM, see [49, 50]. In support of this statement, in Table 3 we measure the maximum l^2-norm quantities $\left|\Pi^{\nabla}_{k,*}\mathcal{D} - \mathcal{I}\right|_2$ and $\left|\Pi^{0}_{k,*}\mathcal{D} - \mathcal{I}\right|_2$ on the elements of each mesh and for each λ value, as an estimate of the approximation error produced by the projectors involved in the computation of the discrete quantities. As done in [51], we denote by $\Pi^{\nabla}_{k,*}$ and $\Pi^{0}_{k,*}$ the matrices containing on each column the coefficients of the projection $\Pi^{\nabla}_{k}\varphi_i$ and $\Pi^{0}_{k}\varphi_i$ respectively of each basis function φ_i of the VEM space V^k_h; moreover, we indicate with $\mathcal{D}$ the matrix formed by the elements $\mathcal{D}_{ij} = \mathrm{dof}_i(m_j)$, with $m_j \in \mathcal{M}_k(E)$ and the operator dof_i introduced in (9). We can observe how the error produced by the projectors increases with the VEM order as expected; this effect can be mitigated by making a different choice for the basis (1), such as the one proposed in [49, 50], but this is out of the context of this work. In conclusion, the optimization process slightly reduces the approximation errors and the condition number of $\mathbb{A}$; this effect is related to the agglomeration operation, which is able to remove most of the small edges and the bad-shaped "long" elements of the original discretization, as detailed in Fig. 5a.

4.2 Network 2 - realistic DFN problem

The second numerical test considers the computation of the hydraulic head distribution in a realistic DFN setting. The network, presented in Fig. 10, is randomly generated inside the box $[0, 1000] \times [-400, 1400] \times [-350, 1200]$ following the strategy proposed in [52], and it is composed by 86 rounded fractures and 159 interfaces.

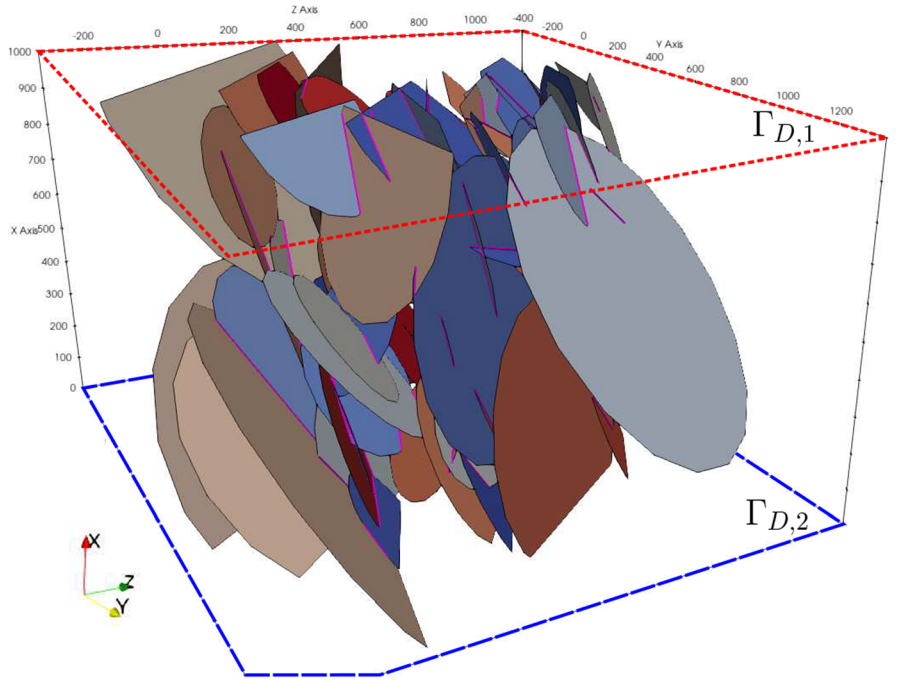


Fig. 10 Network 2; domain description with interfaces S_m highlighted in magenta

We generate two meshes, labeled M1 and M2, by discretizing each fracture F_i with a fixed number of triangular elements (100 and 200, respectively). As in Sect. 4.1, we optimize the meshes through the Quality Agglomeration algorithm of Sect. 2. Figure 11 shows the effects of the agglomeration over a fracture of mesh M1: a lot of small edges and cells are merged into coarser and more regular elements.

Details on the number of elements and DOFs are given in Table 4 and are quite similar to those of Table 2. The number of elements $|\Omega_h|$ decreases by 30% with

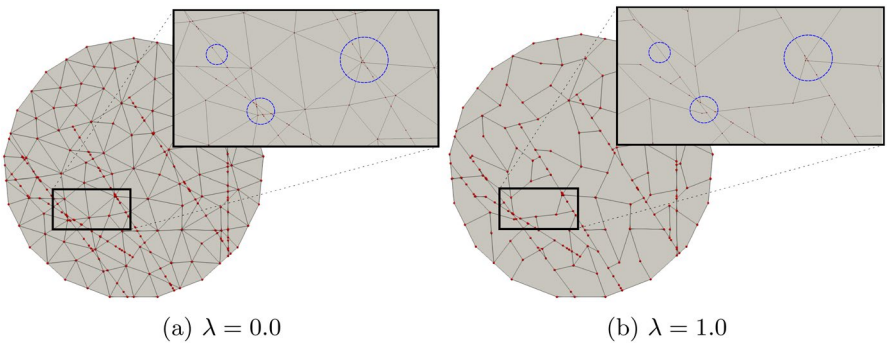


Fig. 11 Network 2, mesh M1; small edges and cells in the original fracture F_3 (**a**) are removed after the optimization process with $\lambda = 1.0$ (**b**). Blue circles compare three example of mesh quality improvements

Table 4 Network 2; analysis of the numerical errors for the different meshes and VEM orders k. λ is the agglomerat ion parameter, $|\Omega_h|$ the number of elements in the mesh, u the discrete solution, $\mathbb{A}$ the stiffness matrix and NNZ its non-zero elements

	λ	$\lvert\Omega_h\rvert$	DOFs	$\lvert\lvert\mu_h\rvert\rvert_{L^2}$	$\lvert\lvert\mu_h\rvert\rvert_{H^1}$	NNZ %	cond ($\mathbb{A}$)
$K = 1$							
Ml	0.00	16,976	11,852	2.3335×10^4	1.8707×10^1	0.08	7.26×10^4
Ml	0.25	11,962	11,807	2.3333×10^4	1.8632×10^1	0.09	7.28×10^4
Ml	1.00	5,950	10,056	2.3324×10^4	1.8443×10^1	0.14	4.83×10^4
M2	0.00	30,968	19,937	2.3324×10^4	1.8676×10^1	0.05	1.08×10^4
M82	0.25	20,865	19,782	2.3323×10^4	1.8615×10^1	0.05	1.08×10^4
M2	1.00	10,355	16,838	2.3322×10^4	1.8478×10^1	0.08	7.21×10^4
$K = 2$							
Ml	0.00	16,976	57,698	2.3331×10^4	1.8493×10^1	0.03	5.90×10^6
Ml	0.25	11,962	47,580	2.3331×10^4	1.8481×10^1	0.04	4.36×10^6
Ml	1.00	5,950	32,053	2.3328×10^4	1.8433×10^1	0.08	3.60×10^6
M2	0.00	30,968	101,852	2.3334×10^4	1.8498×10^1	0.01	1.02×10^7
M2	0.25	20,865	81,336	2.3334×10^4	1.8491×10^1	0.02	7.46×10^6
M2	1.00	10,355	54,428	2.3331×10^4	1.8457×10^1	0.04	2.85×10^6
$K = 3$							
Ml	0.00	16,976	120,520	2.3332×10^4	1.8492×10^1	0.02	1.58×10^{12}
Ml	0.25	11,962	95,315	2.3333×10^4	1.8487×10^1	0.03	1.17×10^{12}
Ml	1.00	5,950	60,000	2.3334×10^4	1.8466×10^1	0.06	1.21×10^{12}
M2	0.00	30,968	214,735	2.3337×10^4	1.8494×10^1	0.01	2.47×10^{12}
M2	0.25	20,865	163,755	2.3337×10^4	1.8491×10^1	0.02	1.81×10^{12}
M2	1.00	10,355	102,373	2.3335×10^4	1.8474×10^1	0.03	2.06×10^{11}

$\lambda = 0.25$ and by 70% with $\lambda = 1.0$; again, the number of DOFs does not change sognificantly in the case $k = 1$, while for $k > 1$ it decreases by 20% and 50% with $\lambda = 0.25$ and 1.0, respectively.

In Table 5 we report the values of the energy functional $\mathcal{E}$ obtained for a generic fracture of the network (F_3, presented in Fig. 11). The results achieved by the optimization algorithm in terms of number of iterations, saved energy, and different energy contributions, are comparable to the ones obtained in Table 1. The measurements suggest that the optimization algorithm is robust with respect to the geometric complexity of the fracture, since the shape and the interfaces of F_3 are highly more complex, compared to those of the fractures in Network 1.

We set the numerical problem by imposing two Dirichlet boundary conditions on the mesh edges which intersect the top and the bottom planes of the box:

$$\begin{cases} u = 0.0 & \text{on } \Gamma_{D,1} := \{(x, y, z) \in \mathbb{R}^3 : x = 1000\} \\ u = 10.0 & \text{on } \Gamma_{D,2} := \{(x, y, z) \in \mathbb{R}^3 : x = 0\} \end{cases}$$

Table 5 Network 2; energy functional over F_3 before ($\mathcal{E}_1$) and after ($\mathcal{E}_2$) the optimization, with the detail of the data cost ($\mathcal{E}_{dc}$) and smoothness cost ($\mathcal{E}_{sc}$) contribution. The percentage in the fourth column is computed with respect to $\mathcal{E}_1$, and in the last two columns with respect to $\mathcal{E}_2$. Column *It* shows the number of iterations needed to converge

		λ	*It*	$(\varepsilon_1 - \varepsilon_2)\%$	$\varepsilon_{2,dc}\%$	$\varepsilon_{2,sc}\%$
M1	0.25	3	1.98×10^4	1.01	11.85	88.15
M1	1.00	6	7.91×10^4	32.93	38.27	61.73
M2	0.25	4	5.80×10^4	1.19	14.89	85.11
M2	1.00	6	2.31×10^4	33 11	32.50	67.50

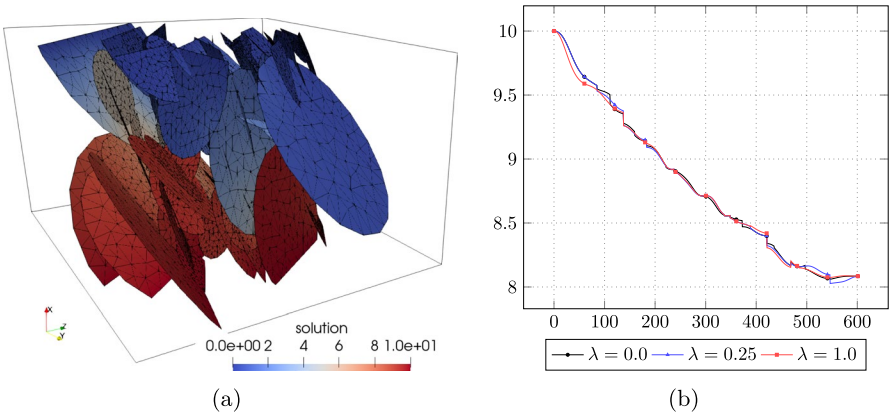


Fig. 12 Network 2; **a** discrete solution u_h for $k = 1$ on mesh M1; **b** u_h for $k = 3$ over a line on fracture F_{49} of mesh M2

We recall that homogeneous Neumann boundary conditions are imposed on the other borders.

The numerical solution for $k = 1$ on the coarser mesh M1 is shown in Fig. 12 a. As no exact solution is available for this network, we measure the L^2-norm and the H^1-seminorm of the discrete solution u_h obtained on the whole network as a quantitative benchmark indicator. Values of $||u_h||_{L^2(\Omega)}$ and $||u_h||_{H^1(\Omega)}$ are reported in Table 4.

In Fig. 12b we perform a qualitative comparison, in which we report the value of u_h (computed for $k = 3$) on a generic line in the middle of the random fracture F_{49} for each optimization value λ. No relevant differences can be identified between the solution on the non-optimized mesh and the one computed on the agglomerated one, despite the huge DOFs reduction (up to 50%) measured in the DOFs column of Table 4.

The last analysis is devoted to the stiffness matrix $\mathbb{A}$ of the discrete problem (10), measuring its non-zero elements and its condition number in Table 4. We can assert that the sparsity pattern of $\mathbb{A}$ is even less influenced by the optimization process than in the test of Sect. 4.1, due to the presence of a higher number of DOFs. We

immediately notice how the condition numbers rapidly increase for higher orders of the method, even faster than in the tests with Network 1. This effect is ascribed to the high complexity of the fracture intersections caused by the randomness of Network 2, which leads to a conforming mesh with very small edges and elongated cells, as highlighted in Fig. 11a. Such pathological elements are partially removed during the agglomeration process, as shown in Fig. 11b and we can observe a remarkable reduction of cond($\mathbb{A}$) after the optimization, see for instance mesh M2 in Table 4c.

To conclude, we show in Table 6 the performance of VEM projectors Π_k^∇ and Π_k^0, measuring the maximum values of the identities which estimate the approximation errors produced by the projection operation. While for VEM order $k = 1$ and $k = 2$ the errors are acceptable, the case $k = 3$ requires further analysis. Thus, in Fig. 13 we compare the distributions of the indicators $\left|\Pi_{3,*}^\nabla \mathcal{D} - \mathcal{I}\right|_2$ and $\left|\Pi_{3,*}^0 \mathcal{D} - \mathcal{I}\right|_2$ on the elements of mesh M2, for the different values of λ. From the reported data we can see how the projection errors remain under control in the vast majority of the mesh, with very few elements (less than 0.1%) with projection error greater than 10^{-5}. As for the previous test, we are confident that those elements are curable by making a different choice for the basis (1). Finally, the plots clearly highlight that the local approximation error of the

Table 6 Network 2; maximum discrepancy of the projection matrices $\Pi_{k,*}^\nabla$ and $\Pi_{k,*}^0$ measured in the mesh elements; note that for $k < 3$ we have $\Pi_{k,*}^\nabla = \Pi_{k,*}^0$

	λ	$\lvert\Pi_{1,*}^\nabla \mathcal{D} - \mathcal{I}\rvert_2$	$\lvert\Pi_{2,*}^\nabla \mathcal{D} - \mathcal{I}\rvert_2$	$\lvert\Pi_{3,*}^\nabla \mathcal{D} - \mathcal{I}\rvert_2$	$\lvert\Pi_{3,*}^0 \mathcal{D} - \mathcal{I}\rvert_2$
M1	0.00	3.9173×10^{-14}	1.8995×10^{-9}	6.0107×10^{-8}	5.1498×10^{1}
M1	0.25	3.9173×10^{-14}	1.8995×10^{-9}	6.5839×10^{-8}	5.1228×10^{1}
M1	1.00	1.1423×10^{-10}	4.6435×10^{-9}	5.7271×10^{-7}	7.9567×10^{1}
M2	0.00	8.6066×10^{-14}	5.5342×10^{-10}	1.1387×10^{-7}	3.9317×10^{2}
M2	0.25	8.6066×10^{-14}	5.5342×10^{-10}	1.1387×10^{-7}	3.9317×10^{2}
M2	1.00	1.4869×10^{-10}	6.1476×10^{-9}	$1.5121. \times 10^{-9}$	2.0541×10^{0}

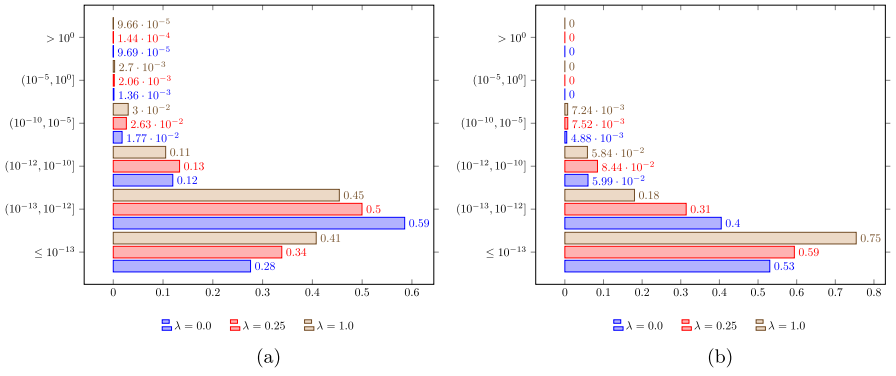


Fig. 13 Network 2, mesh M2; statistic distribution of $\left|\Pi_{3,*}^0 \mathcal{D} - \mathcal{I}\right|_2$ (**a**) and $\left|\Pi_{3,*}^\nabla \mathcal{D} - \mathcal{I}\right|_2$ (**b**), with number of mesh elements reported at the end of each bar

projectors decreases thanks to the optimization process, leading to an improvement of the global condition number and to a more reliable solution.

5 Conclusions

The results obtained with the proposed mesh optimization approach in conjunction with the primal VEM-based formulation applied on DFNs are encouraging. We remark that the Quality Agglomeration strategy presented is performed independently on each fracture of the network, thus its application in larger and more complex DFN configurations is perfectly feasible.

The Quality Agglomeration strategy has proved to be equally effective in both the simple and the complex DFN setting, being able to reduce the total number of mesh elements (up to 65%) and the number of DOFs (up to 50%), while preserving the VEM optimal convergence rates. The former effect leads to a remarkable reduction in the computational effort for the discrete system assembly process; the latter effect translates into a significant gain in terms of computational cost to obtain the numerical solution. Moreover, a slight improvement in the quality of the discretization was observed, both locally on the VEM projector approximation errors measured on each mesh element and globally on the condition number of the stiffness matrix.

A possible limitation of the agglomeration algorithm is that it only compares pairs of elements, i.e. given an element E with neighboring elements E' and E'', the algorithm computes separately the potential quality of $E \cup E'$ and that of $E \cup E''$ not considering the total quality $E \cup E' \cup E''$. Therefore, some small mesh elements still persist in the optimized final mesh, particularly in the neighborhood of fracture intersection segments. This issue could be partially controlled at a higher computational cost by merging the elements after each graph-cut minimization iteration. However, the numerical results presented in this work show that this action is likely not necessary.

In conclusion, we advise the application of the proposed optimization process in the computation of the solution for time-dependent/parametric problems, in which the computational cost reduction would reflect in each time iteration/parameter value resolution.

In addition, we point out that the Quality Agglomeration is modular, in the sense that both the quality indicator and the graph-cut technique can be easily replaced by other objects with the same functionalities. In particular, the quality indicator (2) is peculiar to the VEM, being derived from geometrical assumptions specific to this method. However, we could apply the same approach to other numerical methods (e.g., the Discontinuous Galerkin or the Hybrid High Order methods), starting from the relative geometrical assumptions and deriving new indicators.

Acknowledgements Fabio Vicini and Stefano Berrone have been financially supported by the MIUR project "Dipartimenti di Eccellenza 2018–2022" (CUP E11G18000350001), by PRIN project "Advanced polyhedral discretisations of heterogeneous PDEs for multiphysics problems" 20204LN5N5_003 and by INdAM-GNCS. Tommaso Sorgente, Silvia Biasotti, Gianmarco Manzini and Michela Spagnuolo have been financially supported by the ERC Project CHANGE, which has received funding from the European

Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement no. 694515).

Funding Open access funding provided by Consiglio Nazionale Delle Ricerche (CNR) within the CRUI-CARE Agreement.

References

1. Lo, D.: Finite Element Mesh Generation. CRC Press, New York (2014)
2. Erten, H., Üngör, A., Zhao, C.: Mesh smoothing algorithms for complex geometric domains. In: Proceedings of the 18th International Meshing Roundtable, pp. 175–193. Springer, Albuquerque, NM, USA (2009)
3. Knupp, P.: Introducing the target-matrix paradigm for mesh optimization via node-movement. Engineering with Computers **28**(4), 419–429 (2012)
4. Vartziotis, D., Athanasiadis, T., Goudas, I., Wipper, J.: Mesh smoothing using the geometric element transformation method. Comput. Methods Appl. Mech. Eng. **197**(45–48), 3760–3767 (2008)
5. Alliez, P., Ucelli, G., Gotsman, C., Attene, M.: Recent advances in remeshing of surfaces. Shape Analysis and Structuring, 53–82 (2008)
6. Misztal, M., Bærentzen, J., Anton, F., Erleben, K.: Tetrahedral mesh improvement using multi-face retriangulation. In: Proceedings of the 18th International Meshing Roundtable, pp. 539–555. Springer, Albuquerque, NM, USA (2009)
7. Knupp, P.: Algebraic mesh quality metrics. SIAM J. Sci. Comput. **23**(1), 193–218 (2001)
8. Stimpson, C., Ernst, C., Knupp, P., Pébay, P., Thompson, D.: The Verdict library reference manual. Sandia National Laboratories Technical Report **9**(6) (2007)
9. Chalmeta, R., Hurtado, F., Sacristán, V., Saumell, M.: Measuring regularity of convex polygons. Comput. Aided Des. **45**(2), 93–104 (2013)
10. Zunic, J., Rosin, P.: A new convexity measure for polygons. IEEE Trans. Pattern Anal. Mach. Intell. **26**(7), 923–934 (2004)
11. Huang, W., Wang, Y.: Anisotropic mesh quality measures and adaptation for polygonal meshes. J. Comput. Phys. **410**, 109368 (2020)
12. Sorgente, T., Biasotti, S., Manzini, G., Spagnuolo, M.: The role of mesh quality and mesh quality indicators in the virtual element method. Adv. Comput. Math. **48**(1), 3 (2021)
13. Sorgente, T., Biasotti, S., Manzini, G., Spagnuolo, M.: Polyhedral mesh quality indicator for the virtual element method. Comput. Math. Appl. **114**, 151–160 (2022)
14. Berrone, S., D'Auria, A.: A new quality preserving polygonal mesh refinement algorithm for polygonal element methods. Finite Elem. Anal. Des. **207**, 103770 (2022)
15. Beirão da Veiga, L., Brezzi, F., Cangiani, A., Manzini, G., Marini, L., Russo, A.: Basic principles of virtual element methods. Math. Models Methods Appl. Sci. **23**(01), 199–214 (2013)
16. Goldberg, A., Tarjan, R.: A new approach to the maximum-flow problem. J. ACM **35**(4), 921–940 (1988)
17. Fumagalli, A., Berre, I.: Numerical Methods for Processes in Fractured Porous Media. Birkhäuser, Cham (2019)
18. Martin, V., Jaffré, J., Roberts, J.: Modeling fractures and barriers as interfaces for flow in porous media. SIAM J. Sci. Comput. **26**(5), 1667–1691 (2005)

19. Berrone, S., Pieraccini, S., Scialò, S.: A PDE-constrained optimization formulation for discrete fracture network flows. SIAM J. Sci. Comput. **35**(2), 487–510 (2013)
20. Berrone, S., Grappein, D., Pieraccini, S., Scialó, S.: A three-field based optimization formulation for flow simulations in networks of fractures on nonconforming meshes. SIAM J. Sci. Comput. **43**(2), 381–404 (2021)
21. Burman, E., Hansbo, P., Larson, M., Larsson, K.: Cut finite elements for convection in fractured domains. Comput. Fluids **179**, 726–734 (2019)
22. Köppel, M., Martin, V., Jaffré, J., Roberts, J.: A Lagrange multiplier method for a discrete fracture model for flow in porous media. Comput. Fluids **23**, 239–253 (2019)
23. Berrone, S., Scialò, S., Vicini, F.: Parallel meshing, discretization, and computation of flow in massive discrete fracture networks. SIAM J. Sci. Comput. **41**(4), 317–338 (2019)
24. Berrone, S., Borio, A., Vicini, F.: Reliable a posteriori mesh adaptivity in discrete fracture network flow simulations. Comput. Methods Appl. Mech. Eng. **354**, 904–931 (2019)
25. Chave, F., Di Pietro, D., Formaggia, L.: A hybrid high-order method for Darcy flows in fractured porous media. SIAM J. Sci. Comput. **40**(2), 1063–1094 (2018)
26. Hédin, F., Pichot, G., Ern, A.: A hybrid high-order method for flow simulations in discrete fracture networks. In: Vermolen, F.J., Vuik, C. (eds.) Numerical Mathematics and Advanced Applications ENUMATH 2019, pp. 521–529. Springer, Cham (2021)
27. Beirão da Veiga, L., Lipnikov, K., Manzini, G.: The Mimetic Finite Difference Method, I edn. Modeling, Simulations and Applications, vol. 11. Springer, San Diego, CA, USA (2014)
28. Lipnikov, K., Manzini, G., Shashkov, M.: Mimetic finite difference method. J. Comput. Phys. **257**, 1163–1227 (2014)
29. Antonietti, P., Formaggia, L., Scotti, A., Verani, M., Verzott, N.: Mimetic finite difference approximation of flows in fractured porous media. ESAIM: Math. Model. Numer. Anal. **50**(3), 809–832 (2016)
30. Brezzi, F., Manzini, G., Marini, D., Pietra, P., Russo, A.: Discontinuous Galerkin approximations for elliptic problems. Numer. Methods Partial Differ. Equ. **16**(4), 365–378 (2000)
31. Fumagalli, A., Keilegavlen, E.: Dual virtual element method for discrete fractures networks. SIAM J. Sci. Comput. **40**(1), 228–258 (2018)
32. Benedetto, M., Berrone, S., Pieraccini, S., Scialò, S.: The virtual element method for discrete fracture network simulations. Comput. Methods Appl. Mech. Eng. **280**, 135–156 (2014)
33. Benedetto, M., Berrone, S., Scialò, S.: A globally conforming method for solving flow in discrete fracture networks using the virtual element method. Finite Elem. Anal. Des. **109**, 23–36 (2016)
34. Berrone, S., Borio, A., D'Auria, A.: Refinement strategies for polygonal meshes applied to adaptive VEM discretization. Finite Elem. Anal. Des. **186**, 103502 (2021)
35. Adams, R.A., Fournier, J.J.F.: Sobolev Spaces, 2nd edn. Pure and Applied Mathematics. Academic Press, Amsterdam (2003)
36. Cangiani, A., Manzini, G., Sutton, O.: Conforming and nonconforming virtual element methods for elliptic problems. IMA J. Numer. Anal. **37**(3), 1317–1354 (2016)
37. Beirão da Veiga, L., Vacca, G.: Sharper error estimates for virtual elements and a bubble-enriched version. SIAM J. Numer. Anal. **60**(4), 1853–1878 (2022)
38. Sorgente, T., Prada, D., Cabiddu, D., Biasotti, S., Patane, G., Pennacchio, M., Bertoluzza, S., Manzini, G., Spagnuolo, M.: 1. VEM and the mesh. SEMA SIMAI Springer series, vol. 31, pp. 1–54. Springer, Nature Switzerland AG (2021). ISBN: 978-3-030-95318-8
39. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. IEEE Trans. Pattern Anal. Mach. Intell. **23**(11), 1222–1239 (2001)
40. Kolmogorov, V., Zabin, R.: What energy functions can be minimized via graph cuts? IEEE Trans. Pattern Anal. Mach. Intell. **26**(2), 147–159 (2004)
41. Boykov, Y., Kolmogorov, V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Trans. Pattern Anal. Mach. Intell. **26**(9), 1124–1137 (2004)
42. Veksler, O., Delong, A.: Multi-label optimization. https://vision.cs.uwaterloo.ca/code/ (2010)
43. Ahmed, E., Jaffré, J., Roberts, J.: A reduced fracture model for two-phase flow with different rock types. Mathematics and Computers in Simulation 137, 49–70 (2017). MAMERN VI-2015: 6th International Conference on Approximation Methods and Numerical Modeling in Environment and Natural Resources
44. Dershowitz, W., Fidelibus, C.: Derivation of equivalent pipe network analogues for three-dimensional discrete fracture networks by the boundary element method. Water Resour. Res. **35**(9), 2685–2691 (1999)

45. Ridgway Scott, L., Brenner, S.: The Mathematical Theory of Finite Element Methods, 3rd edn. Texts in Applied Mathematics 15. Springer, New York (2008)
46. Beirão da Veiga, L., Lovadina, C., Russo, A.: Stability analysis for the virtual element method. Math. Models Methods Appl. Sci. **27**(13), 2557–2594 (2017)
47. Beirão da Veiga, L., Brezzi, F., Marini, L., Russo, A.: Virtual element method for general second-order elliptic problems on polygonal meshes. Math. Models Methods Appl. Sci. **26**(4), 729–750 (2016)
48. Guennebaud, G., Jacob, B.: Eigen v3. http://eigen.tuxfamily.org (2010)
49. Mascotto, L.: Ill-conditioning in the virtual element method: stabilizations and bases. Numer. Methods Partial Differ. Equ. **34**(4), 1258–1281 (2018)
50. Berrone, S., Borio, A.: Orthogonal polynomials in badly shaped polygonal elements for the virtual element method. Finite Elem. Anal. Des. **129**, 14–31 (2017)
51. Beirão da Veiga, L., Brezzi, F., Marini, L., Russo, A.: The hitchhiker's guide to the virtual element method. Math. Models Methods Appl. Sci. **24**(8), 1541–1573 (2014)
52. Srinivasan, S., Hyman, J., Karra, S., O'Malley, D., Viswanathan, H., Srinivasan, G.: Robust system size reduction of discrete fracture networks: a multi-fidelity method that preserves transport characteristics. Comput. Geosci. **22**(6), 1515–1526 (2018)

Authors and Affiliations

Tommaso Sorgente[1] · Fabio Vicini[2,3] · Stefano Berrone[2,3] · Silvia Biasotti[1] · Gianmarco Manzini[1,3] · Michela Spagnuolo[1]

✉ Tommaso Sorgente
tommaso.sorgente@ge.imati.cnr.it

Fabio Vicini
fabio.vicini@polito.it

Stefano Berrone
stefano.berrone@polito.it

Silvia Biasotti
silvia.biasotti@ge.imati.cnr.it

Gianmarco Manzini
marco.manzini@imati.cnr.it

Michela Spagnuolo
michela.spagnuolo@ge.imati.cnr.it

[1] Istituto di Matematica Applicata e Tecnologie Informatiche 'E. Magenes', Consiglio Nazionale Delle Ricerche, Genoa, Italy

[2] Dipartimento di Scienze Matematiche 'G. L. Lagrange', Politecnico di Torino, Turin, Italy

[3] INdAM Research Group GNCS, Verona, Italy