AGU PUBLICATIONS

Water Resources Research

RESEARCH ARTICLE

10.1002/2015WR017729

**Key Points:**

- 3-D upscaling of naturally fractured reservoir
- Nonmatching grids using embedded representation of fractures
- Dual porosity-dual permeability framework

**Correspondence to:**
A. Fumagalli,
alessio.fumagalli@uib.no

**Citation:**
Fumagalli, A., L. Pasquale, S. Zonca, and S. Micheletti (2016), An upscaling procedure for fractured reservoirs with embedded grids, *Water Resour. Res.*, *52*, 6506–6525, doi:10.1002/2015WR017729.

Received 1 AUG 2015
Accepted 31 JUL 2016
Accepted article online 5 AUG 2016
Published online 25 AUG 2016

# An upscaling procedure for fractured reservoirs with embedded grids

**Alessio Fumagalli[1,2], Luca Pasquale[3], Stefano Zonca[1], and Stefano Micheletti[1]**

[1]Dipartimento di Matematica, Politecnico di Milano, Milan, Italy, [2]Currently at Department of Mathematics, University of Bergen, Bergen, Norway, [3]MOXOFF S.P.A., Milan, Italy

**Abstract** Upscaling of geological models for reservoir simulation is an active and important area of research. In particular, we are interested in reservoirs where the rock matrix exhibits an intricate network of fractures, which usually acts as a preferential path to the flow. Accounting for fractures' contribution in the simulation of a reservoir is of paramount importance. Here we have focused on obtaining effective parameters (e.g., transmissibility) on a 3-D computational grid on the reservoir scale, which account for the presence, at a finer spatial scale, of fractures and a network of fractures. We have essentially followed the idea illustrated in Karimi-Fard et al. (2006), yet this work has some notable aspects of innovation in the way the procedure has been implemented, and in its capability to consider rather general corner-point grids, like the ones normally used in reservoir simulations in the industry, and complex and realistic fracture networks, possibly not fully connected inside the coarse cells. In particular, novel contribution is the employment of an Embedded Discrete Fracture Model (EDFM) for computing fracture-fracture and matrix-fracture transmissibilities, with a remarkable gain in speedup. The output is in the form of transmissibility that, although obtained by considering single-phase flow, can be used for coarse-scale multiphase reservoir simulations, also via industrial software, such as Eclipse, Intersect, or GPRS. The results demonstrate the effectiveness and computational efficiency of the numerical procedure which is now ready for further testing and industrialization.

## 1. Introduction

Reservoir simulation is widely applied in the petroleum industry for the prediction and management of reservoir performance. A primary input to the flow simulator is the geological description of the reservoir, which is typically provided in the form of a high-resolution geocellular model containing petrophysical (e.g., porosity and permeability) data [*Durlofsky*, 2003]. However, the high-level resolution of this model exceeds the capabilities of standard reservoir simulators, and "upscaling" or "scale up" techniques are required to coarsen the reservoir description to scales more suitable for flow computation. A key issue with any upscaling procedure is how well the coarsened (upscaled) model replicates important aspects of the fine-scale flow behavior, such as total injection or production rate, average pressure or saturation throughout the reservoir, and breakthrough times of injected fluids.

In particular, we are interested in fractured reservoirs, i.e., reservoirs characterized by thousands of intersecting fractures. There are, however, a large number of challenges associated with predicting the flow through fractured systems. An example is the development of reliable and efficient mathematical models to describe the fracture networks from their geological representation and the flow through those networks. Actually, naturally fractured reservoirs exhibit fractures on several scales, which can also be heterogeneous across these scales and even on each individual scale, so that it is not possible, in general, to associate one distinct porosity with each fracture. A general approach to deal with such an issue is to first provide a means to cluster fractures at different scales. For example, equivalent homogeneous Dual-Porosity (DP) systems [*Warren and Root*, 1963], in the spirit of a dichotomy, subdivide the reservoir in matrix and fractures, because of their different fluid storage and conductivity characteristics. In this work, we assume that the size of the fractures is homogeneous. Since the permeability of the fractures is typically orders of magnitudes greater than that of the rock matrix, water, oil, and gas flow preferably in networks of fractures. An efficient and effective modeling of fluid flows in fractured media is a complex task due to this highly heterogeneous nature of the flow processes.

On the other hand, traditional homogenization techniques are not suitable for fractured systems because the spatial scales are not highly separated, and the topology of fractures may strongly affect the apparent (upscaled) permeability. A numerical upscaling technique based on the numerical solution of the fracture network and matrix flow at a "local" fine scale is preferable. The fluid that flows in fractures, matrix, and between matrix and fractures has different characteristics. Moreover, fractures distribution in subsurface formations usually displays significant variations in connectivity and size over the formation. Large and strongly connected fractures are typically located near bedding planes and fault zones, while small and disconnected fractures are usually located away from those regions. The variation in fracture properties, especially fracture connectivity, requires modeling different fracture zones using different numerical treatments to achieve sufficiently accurate upscaling results. Since fracture permeability is extremely high in comparison with matrix permeability, a reasonable assumption is that the flow in fractures reaches a pseudo steady state (constant rate of change of pressure) just after the global flow starts. Transfer functions or shape factors [*van Heel et al*., 2008] can thus be derived to couple the fluid flow in the matrix and fractures based on the fracture characterization and are used to propagate the fine-scale information to the coarse-scale reservoir simulation. The assumption of instantaneous pseudo steady state is not valid if the coarse grid block is large, or the matrix permeability is small, which is usually the case in field-scale reservoir simulations.

The Discrete Fracture Model (DFM) represents one of the most accurate methodologies for accurately describing flow in fracture networks as it entails the direct numerical simulation of flow through the fractured porous media. Using DFM, the rock matrix and fractures are represented explicitly and Darcy flow equations are solved. However, one of the major drawbacks of this method is its high computational cost. Nevertheless, there has been an increasing interest of the scientific community as well as of the reservoir engineering community in DFM as a result also of the availability of more powerful computers and more effective discretization techniques [*Li and Lee*, 2008; *Karimi-Fard et al*., 2004]. Although DFMs are becoming more and more efficient, the application of these methods at field scale is not currently realistic. Yet they are a valid tool to perform a numerical upscaling.

*Pruess and Narasimhan* [1985], *Pruess* [1992], *Gong et al*. [2006, 2008b, 2008a], and *Tatomir et al*. [2011], applied also in *Hui et al*. [2008, 2013], introduced a systematic upscaling methodology that constructs a generalized DP model from fine-scale discrete fracture characterizations. In particular, the Multiple INteracting Continua (MINC) method, introduced in *Pruess and Narasimhan* [1985] and *Pruess* [1992], is an approximate method for modeling fluid and heat flow in fractured-porous media, which extends the dual porosity concept. It is based on the property that fractures have large permeability and small porosity (when averaged over a reservoir subdomain), while the intact rock (the rock "matrix") has the opposite characteristics. Thus, any disturbance in reservoir conditions propagates fast through the network of interconnected fractures, while it affects the matrix blocks only slowly. According to the resulting model, referred to as "sugar-cube," the global flow in a fractured-porous medium occurs exclusively through the fractures, while the matrix blocks may exchange fluid locally with the fractures. Later, the MINC method has been improved in *Gong et al*. [2006, 2008b] and *Tatomir et al*. [2011]. This technique, referred to as Multiple Sub-Region (MSR) method, introduces local subregions to resolve dynamics within the matrix and provides appropriate coarse-scale parameters that describe fracture-fracture, matrix-fracture, and matrix-matrix flow. Unlike the DP approach, in which the number of subregions inside each coarse block is reduced to one in addition to that accounting for the fractures, the MSR approach can be viewed as a generalization. In fact, several transmissibility terms are computed to represent both the interaction between fracture and matrix, and the dynamic inside the matrix, whereas, in the DP method, only one term related to fracture-fracture (block-to-block) and one term related to fracture-matrix (within block) flow is computed. The MSR approach provides a more realistic characterization of the flux exchange between matrix and fractures since the construction of the subregions reflects the actual fracture distribution. Moreover, with more than one subregion in each coarse block, MSR is capable of modeling more accurately the effects of transients inside the matrix. The upscaled properties, in particular the transmissibility, are computed by solving locally a continuity equation for the pressure, obtained by plugging Darcy's law into the mass conservation equation. Extensions of the aforementioned works are discussed in *Hui et al*. [2007] and *Gong et al*. [2011], where additional connections between the subregions of the coarse cells are considered leading to a Dual-Porosity/Dual-Permeability (DPDK) model.

With regard to complex three-dimensional problems, a computational grid conforming to the fractures can be very difficult to generate, due to the small angles that may occur at the intersection of two or more fractures. Moreover, many very small elements could be created which could badly affect the numerical solution. Finally, since the typical grids used in commercial reservoir software are the so-called corner-point grids, it is advantageous employing this type of mesh in order to easily interface with these programs. Thus, in order not to build a conforming mesh and to alleviate the computational cost, we rely on the methodology first introduced by *Li and Lee* [2008] and expanded in *Panfili et al*. [2013], *Moinfar et al*. [2014], *Panfili and Cominelli* [2014], and *de Araujo Cavalcante Filho et al*. [2015]. This method, known as Embedded Discrete Fracture Model (EDFM), allows one to generate the fractures mesh and porous medium grid independently, thus avoiding the need for a conforming mesh and allowing one to use corner-point grids.

In this work, we have developed an accurate description of fracture networks to obtain an upscaled representation of the porous medium system through the use of an MSR technique. We have addressed the three-dimensional problem directly and considered corner-point grids, typically met in reservoir simulations. The aim was to enable the computation of flow in the upscaled grid (at reservoir scale) by the computation of appropriate transmissibility between fracture and matrix, among fractures, and among different matrix blocks. This has required solving different methodological and implementation issues, which are described in the following sections. In more detail, our original contribution is the employment of EDFM to compute the fracture-fracture, matrix-fracture, and matrix-matrix transmissibilities in every coarse cell, with a considerable saving on the computational cost, due to the use of a Cartesian mesh that is built at run time. Another novelty of our approach is the possibility to deal with locally disconnected fractures, i.e., the fractures in a given coarse cell or pair of coarse cells need not be fully connected. We point out that also all of the algorithms that we have devised for detecting intersections between objects of different geometric nature, e.g., a parametric bilinear surface and a line, are thoroughly original and much of our effort has been focused on obtaining robust floating-point geometric predicates.

This paper starts with an introduction to the Darcy equations and to their numerical discretization through a two-point finite volume scheme in section 2. The EDFM method is recalled in section 3, along with its efficient and effective implementation. Section 4 describes the upscaling procedure and the special features which we have devised, i.e., the multiple subregion approach, and the treatment of the communication between two adjacent cells. A numerical assessment of the whole procedure is provided in section 5. In particular, we address (a) the comparison between the computational cost to build a constrained 3-D Delaunay mesh and the EDFM method for a sample of fractures and (b) the comparison between the proposed approach and a standard software used in oil industries. Finally, we apply the upscaling procedure to a realistic 3-D case in order to study the production oil for a two-phase flow problem. Some conclusions and an outlook for future developments are gathered in section 6.

## 2. Governing Equations

Let $\Omega \subset \mathbb{R}^N$, for $N=2$ or 3, be the physical domain, which represents the whole reservoir. To ease the presentation, we require that $\Omega$ be a regular domain with Lipschitz boundary, denoted by $\partial\Omega$ with unit outward normal $\boldsymbol{n}_{\partial\Omega}$. The reservoir is considered as a fully saturated porous medium. We suppose that it is possible to define in $\Omega$ the Representative Elementary Volume (REV) [see, e.g., *Bear*, 1972], such that the Darcy equation can be applied to describe the fluid flow inside the domain. We are interested in computing the steady pressure field $p$ (Pa) and the Darcy velocity (or the flux) $\boldsymbol{u}$ (m/s) in the whole domain $\Omega$ governed by the following Darcy system of equations:

$$\begin{cases} \boldsymbol{u}+\dfrac{\Lambda}{\mu}\nabla p=\boldsymbol{0} & \text{in } \Omega \\ \nabla\cdot\boldsymbol{u}=q & \\ p=0 & \text{on } \partial\Omega \end{cases}, \qquad (1)$$

where $q$ is a source/sink term, $\Lambda$ (m$^2$) is the permeability matrix, which is a symmetric and positive definite tensor, i.e., $\Lambda \in \mathbb{R}^{N\times N}$, and $\mu$ (Pa s) is the viscosity of the liquid. For simplicity, we have supposed that system (1) is supplemented with Dirichlet boundary conditions to ensure its well posedness (see, e.g., *Brezzi and Fortin* [1991] for a detailed analysis of this problem). We have chosen a numerical scheme that requires recasting system (1) in its primal formulation, where only the pressure field $p$ is considered as an unknown. Instead of considering system (1), we thus solve the following problem:

$$\begin{cases} -\nabla \cdot \dfrac{\Lambda}{\mu} \nabla p = q & \text{in } \Omega \\ p = 0 & \text{on } \partial\Omega \end{cases}. \tag{2}$$

### 2.1. Numerical Approximation

From the numerical viewpoint, we employ the classical two-point finite volume scheme to solve the problem (see *Eymard et al*. [2000] for a detailed description). This is justified by its local mass conservation property and by the resulting M-matrix property of the stiffness matrix. Moreover, it can be easily used to discretize the diffusive term of single-phase as well as multiphase flows, and it is one of the fastest scheme available in the literature for solving diffusive partial differential equations. We introduce the scheme for a general grid cell, then the application to a particular problem is straightforward. For each cell $K$ of the computational grid, the scheme can be written as

$$\sum_{f \in \mathcal{F}_K} F_{f,K}(p) = \int_K q \, \mathrm{d}\boldsymbol{x},$$

where $\mathcal{F}_K$ is the set of facets (edges in 2-D and faces in 3-D) $f$ of the cell $K$, and $F_{f,K}$ ($\mathrm{m}^3/\mathrm{s}$) is the numerical flux related to the diffusive operator across the facet $f$. Given two adjacent cells, $K$ and $L$, sharing the facet $f$, we enforce the local conservation of mass through $f$: $F_{f,K}(p) + F_{f,L}(p) = 0$. The two-point flux scheme approximates the flux in the following way:

$$F_f(p) := \frac{T_{KL}}{\mu}(p_K - p_L), \tag{3}$$

where $p_K$ and $p_L$ are the pressures in the cells $K$ and $L$, respectively. Since $T_{KL} = T_{LK}$, with this definition, the conservation of mass is automatically satisfied. In (3), we have introduced the transmissibility $T_{KL}$ ($\mathrm{m}^3$) between adjacent cells, which is the main ingredient of the two-point flux approximation scheme. The computation of the transmissibility is based on some geometrical quantities of the two cells $K$ and $L$, as well as on the permeability tensor. The transmissibility can be computed through the following formula:

$$T_{KL} = \frac{T_K T_L}{T_K + T_L}, \tag{4}$$

where $T_K$ and $T_L$ are the so called "half transmissibilities" related to the cells $K$ and $L$, respectively. These objects can be computed by

$$T_K := \frac{(\boldsymbol{x}_K - \boldsymbol{x}_f)^\top}{||\boldsymbol{x}_K - \boldsymbol{x}_f||_2^2} \Lambda \boldsymbol{n}_{K,f} |f|, \tag{5}$$

with $\boldsymbol{n}_{K,f}$ the unit outward normal to $f \in \mathcal{F}_K$, $\boldsymbol{x}_K$ the center of gravity of the cell $K$, and $\boldsymbol{x}_f$ the center of gravity of the facet $f$.

### 2.2. Virtual Fracture Cells Method

Problem (2) can be applied also to describe the flow in the fractures. We suppose that the fractures have an aperture $a$ ($m$) of several orders of magnitude smaller than their other characteristic sizes and porous medium grid size. Moreover, we assume that the permeability of the fractures may be several orders of magnitude different from the permeability of the surrounding porous medium. Due to the first assumption, from a geometrical point of view, we represent the fractures by objects of codimension one: surfaces for $N = 3$ or lines for $N = 2$. A commonly used approach in the approximation via DFM is based on the construction of an additional mesh consisting of *virtual fracture cells* to discretize the problem in each fracture (see *Faille et al*. [2016]).

The virtual fracture cells are obtained by extrusion of fracture facets along their normal direction by a thickness $\pm a/2$. The global mesh is obtained by the union of the mesh which discretizes the rock matrix and the mesh comprising the virtual fracture cells. This approach allows us to treat the elements of the fracture discretization as any other cell, so that we do not need a special treatment for the fracture cells. In practice, the virtual fracture cells are never constructed explicitly, but only introduced to compute the geometric quantities required by the two-point approximation.

Another approach is to consider the reduced model approximation described in *Martin et al*. [2005], *Angot et al*. [2009], and *D'Angelo and Scotti* [2012]. Again, we assume that the fractures are represented by objects of codimension one, but in this case, the construction of the tangential operators and the imposition of suitable coupling conditions between the porous matrix and the fractures are required. However, the implementation of the two approaches does provide the same results.

## 3. Embedded Discrete Fracture Model

To solve problem (2) by a two-point finite volume scheme, a discretization of both porous medium and fractures is required. We focus our attention on complex three-dimensional problems. A representation of the computational grid with a conforming mesh that adapts to the fractures may be very complex to generate, due to the small angles that may occur at the intersection of two or more fractures. Moreover, this procedure could generate many very small elements that could badly affect the numerical solution. Finally, since the typical grids used in the commercial reservoir software are the so-called corner-point grids, it is convenient to employ such kind of grid in order to easily interface with these programs.

In order to avoid having to build a conforming mesh and to alleviate the computational cost, we rely on the methodology first introduced by *Li and Lee* [2008] and expanded in *Panfili et al*. [2013], *Moinfar et al*. [2014], *Panfili and Cominelli* [2014], and *de Araujo Cavalcante Filho et al*. [2015]. This method, known as Embedded Discrete Fracture Model (EDFM), allows one to generate the fractures mesh and porous medium grid independently, thus avoiding the need for a conforming mesh and allowing one to use corner-point grids.

Fractures are meshed so that each corner-point cell that they intersect contains one and only one fracture cell per fracture. We consider only the case of planar quadrilateral fractures but parameterized as bilinear surfaces to exploit the same intersection algorithm, presented in Appendix A, used for dealing with the faces of a general corner-point element. Intersections between fractures are computed, but do not affect the meshing strategy of either fractures or corner-point grid. They are instead used to compute transmissibilities between different fracture cells (see, e.g., Figure 1, bottom right).

A degree of freedom is assigned to each matrix cell and to each fracture cell. This means that transmissibilities between matrix and fracture cells, as well as those between different fracture cells need to be computed.

### 3.1. Meshing Algorithm

The first step toward computing all transmissibilities is to actually mesh the fractures. This implies identifying the corner point cells intersected by each fracture, and computing the polygons resulting from the intersection of the fracture with each cell.

The vertices $V$ of each polygon are the union of three sets of points: fracture vertices lying inside the element $V_i$, intersections between an element face and a fracture edge $V_f$, and intersections between an element edge and the fracture $V_e$ (see, e.g., Figure 1, top right)

$$V = V_i \cup V_f \cup V_e.$$

The second and third set of points, $V_f$ and $V_e$, can be found by computing the intersections between a line segment, the fracture edge, or the element edge, respectively, and a bilinear surface, the element face, or the fracture. Computing these intersections gives rise to a system of nonlinear equations. In Appendix A, we present an algorithm that provides an explicit solution to this system, avoiding the need for iterative methods, while taking special care of numerical errors and floating-point arithmetic.

Last, testing whether a fracture vertex lies inside a trilinear element, i.e., finding $V_i$, can be done using a ray-casting algorithm. For a detailed presentation [see *Shimrat*, 1962, Algorithm 112]. In particular, we can build a line segment going from the element parametric center to the point whose position we want to check. We can then count the total number of intersections between this segment and each element face, again, by finding the intersection between a line and a bilinear surface. An odd number of intersections means

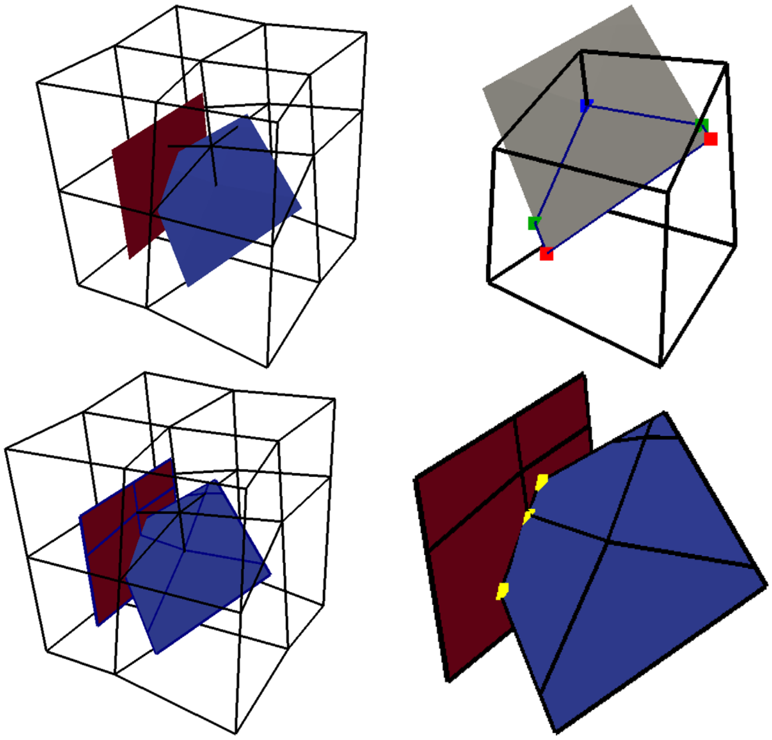

**Figure 1.** Example of meshing using EDFM. (top left) The two input intersecting fractures and the corner-point grid (2×2×2). Only intersections between the fractures and between each fracture with the coarse cell need to be computed. (top right) The three types of points found by intersecting a fracture, in gray, with a corner-point cell: $V_i$ in red, $V_f$ in green, and $V_e$ in blue. The resulting fracture cell is the one with the blue edges. (bottom left) The resulting fracture meshes. (bottom right) The two intersection segments (with yellow endpoints) between the fracture cells.

that the point is outside the element, while an even number indicates that it is inside, as stated by the ray-casting algorithm.

Once we have the set of vertices $V$, the corresponding fracture cell polygon can be built by ordering the vertices clockwise (or anticlockwise) around their midpoint (computed as their average).

To reduce the number of corner-point cells to be tested for intersections with a specific fracture, a search structure on the corner-point grid can be used. For this purpose, following the idea presented in *Formaggia* [1998], we chose an ADTree, which, given the bounding box of a fracture, provides a possible list of cells that it intersects.

The intersections between different fracture cells, required to compute transmissibilities, are found by intersecting the edges of one fracture with the surface of the other. Since we already know in which corner-point cell each fracture cell falls, we can restrict the computation of intersections to those in the same corner-point cell, thus greatly reducing the computational cost.

*Remark 1*. Because corner point cells are represented as trilinear elements, their faces are bilinear surfaces. Actually, their intersections with a fracture are therefore nonlinear. In order to avoid having to deal with complex geometric objects and to be able to use simple closed-formed formulas to compute transmissibilities, these intersections are, however, approximated by straight line segments.

### 3.2. Computation of Transmissibilities

Once we have defined all fracture cells and found the intersections between them, we can proceed to compute the transmissibilities. Let us denote with $A_{\Sigma K}$ the surface area of the fracture cell $\Sigma$ contained in a matrix block $K$, with $a_\Sigma$ its aperture and with $\boldsymbol{n}_\Sigma$ the unit normal vector to the fracture surface. The transmissibility between a matrix cell and a fracture cell contained in it is computed using the formula proposed in *Li and Lee* [2008]

$$T_{K\Sigma}=A_{\Sigma K}\frac{\boldsymbol{n}_\Sigma^\top\Lambda\cdot\boldsymbol{n}_\Sigma}{d_{K\Sigma}},$$

where $d_{K\Sigma}$ denotes the average distance between a generic point of the fracture cell and the fracture plane. The transmissibility between two fracture cells, $\Sigma$ and $\Theta$, is instead computed similarly to (4)

$$T_{\Sigma\Theta}=\frac{T_\Sigma T_\Theta}{T_\Sigma+T_\Theta}. \tag{6}$$

First we consider two fracture cells belonging to the same fracture and sharing the edge $\sigma$. However, $\sigma$ is the representation of two distinct edges, $\sigma_\Theta$ and $\sigma_\Sigma$, with their own normals $\boldsymbol{n}_{\sigma_\Theta}$ and $\boldsymbol{n}_{\sigma_\Sigma}$, respectively. The half transmissibilities, $T_\Sigma$ and $T_\Theta$, are computed locally and are given by the formula derived from the two-point flux approximation found in *Karimi-Fard et al*. [2004]

$$T_\Theta=\frac{|\sigma_\Theta|\lambda_\Theta a_\Theta(\boldsymbol{x}_{\sigma_\Theta}-\boldsymbol{x}_\Theta)\cdot\boldsymbol{n}_{\sigma_\Theta}}{d_{\Theta\sigma_\Theta}||\boldsymbol{x}_{\sigma_\Theta}-\boldsymbol{x}_\Theta||_2}.$$

A similar formula can be provided for $T_\Sigma$. In the previous formula, $\lambda_\Theta,\ a_\Theta$, and $\boldsymbol{x}_\Theta$ are the permeability, aperture, and center of gravity, respectively, of fracture cell $\Theta$, $\boldsymbol{x}_{\sigma_\Theta}$ is the center of gravity of $\sigma_\Theta$, and $d_{\Theta\sigma_\Theta}$ is the average distance between a generic point of $\Theta$ and $\sigma_\Theta$, computed by

$$d_{\Theta\sigma_\Theta}=\frac{1}{|\Theta|}\int_\Theta||\boldsymbol{x}_{\sigma_\Theta}-\boldsymbol{x}||_2\mathrm{d}\boldsymbol{x}.$$

A similar formula can be used for $\Sigma$.

For a generic fracture cell $\Sigma$ belonging to different fractures and intersecting each other on line segment $s$, the half transmissibility is instead computed as

$$T_\Sigma=\frac{|s|\lambda_\Sigma a_\Sigma}{d_{\Sigma s}},$$

where $d_{\Sigma s}$ is the average distance of a generic point of $\Sigma$ to the line to which $s$ belongs.

EDFM has multiple advantages that come in handy to make the MSR upscaling methodology as robust and fast as possible. On the one hand, its ability to handle complex fracture geometries allows us to handle any configuration. On the other hand, its speed in generating meshes and computing transmissibilities makes it particularly suitable to be used in a scenario like that of Multiple Sub-Regions upscaling, where a multitude of problems need to be solved, and where it can also help in identifying fracture networks, i.e., all those fracture cells that are somehow connected one to another.

## 4. Upscaling Procedure

In the case of applications, where thousands of fractures are considered, it is very costly to construct a conforming mesh with a good quality to approximate and solve problem (2) with the virtual fracture cells. Moreover, due to the uncertainty in the underground, it can be necessary to perform a multiple scenario analysis by changing the fractures position, thus making it unreasonable and unaffordable to build a different mesh for each realization. For these reasons, we have chosen a different strategy to solve the global problem, considering an upscaling technique to derive effective model parameters, at a coarser scale, which accurately represent the fine scale model. An exhaustive presentation of the classical upscaling techniques, both numerical and analytical, in unfractured porous media, can be found in *Durlofsky* [2003]. Nevertheless, in this work we focus our attention on fractured reservoirs. The main idea of the numerical upscaling is to consider a global coarse grid $\mathcal{G}$ which describes the upscaled reservoir and that it is geometrically independent of the domain's fractures $\Omega_f$. Typically, $\mathcal{G}$ is a corner-point grid, which represents the different sedimentary layers of the reservoir. The upscaling procedure computes the following upscaled properties: the transmissibility between adjacent coarse blocks, the depth, and the porous volume of each coarse block. However, the methods described in *Durlofsky* [2003], as many other techniques proposed in the literature, do not explicitly consider the presence of the fractures. For this reason, we have considered the upscaling procedure proposed in *Karimi-Fard et al*. [2006], *Gong et al*. [2006], and *Tatomir et al*. [2011].

To ease the presentation we recall the basic ideas of the upscaling scheme, extended by adding a matrix-matrix connection between coarse cells.

### 4.1. Multiple Subregions

In this section, we present the upscaling procedure proposed by *Karimi-Fard et al.* [2006] and *Tatomir et al.* [2011]. Considering two adjacent coarse cells of the global mesh $\mathcal{G}$, the main assumption of the scheme is that the flow across the two cells takes place only through the fractures networks and not through the porous medium. This hypotheses is motivated by the fracture permeability, which is several orders of magnitude greater than the permeability of the rock matrix. Moreover, we assume that the interchange of flow between the fractures and the rock matrix is localized only inside each coarse cell. The method of multiple subregions, introduced in the aforementioned work, considers a subdivision of each coarse cell to enhance the description of the porous medium and to obtain better results. The resulting degrees of freedom (d.o.f.) of the coarse model are associated with each coarse cell and represent the pressure inside the local fractures networks and the pressure in each subregion of the rock matrix. With "local network," we refer to the set of fractures that are connected inside a specific cell or pair of cells. Thus, in general, inside a cell or a pair of cells, we can distinguish several local networks, one for each group of connected fractures. Following the idea proposed in *Karimi-Fard et al.* [2006], the upscaled model could be sketched with a graph, where each node represents a degree of freedom, and each link represents the connection, i.e., the transmissibility, between the degrees of freedom. For example, in *Karimi-Fard et al.* [2006, Figure 2], the graph represents five coarse cells in which each of them contains a system of fractures and three porous matrix subregions. The communication between the coarse cells occurs only through the fractures.

In the sequel, to compute the upscaled quantities we solve local problems involving single cells, or pairs of cells. Considering a general coarse block $K \in \mathcal{G}$, we need to build a finer grid, indicated by $\mathcal{M}_K$, consisting of $C_K$ polyhedra in 3-D, polygons in 2-D, or briefly cells, indicated by $k$, such that

$$\bar{K} = \overset{C_K}{\underset{i=1}{\cup}} \overline{k_i},$$

which is used to solve the local problems. In the case when two adjacent coarse blocks $K, L \in \mathcal{G}$ are involved in the computation, we construct the fine mesh, indicated by $\mathcal{M}_{KL}$, as the union of each submesh $\mathcal{M}_{KL} = \mathcal{M}_K \cup \mathcal{M}_L$. The interface $\Gamma_{KL}$ between $K$ and $L$ is then honored by the discretization and, for simplicity, we require that the resulting mesh be conforming. In each pair of coarse cells, it is possible to identify $N_{KL}$ local networks of fractures. We indicate by $\mathcal{N}^i_{KL}$ the mesh associated with the discretization of the $i$th local network of fractures inside the pair of cells $K$ and $L$. The global set of fractures for pair of cells is denoted by $\mathcal{N}_{KL}$. We have

$$\mathcal{N}_{KL} = \overset{N_{KL}}{\underset{i=1}{\cup}} \mathcal{N}^i_{KL} \quad \text{and} \quad \mathcal{N}^i_{KL} \cap \mathcal{N}^j_{KL} = \emptyset \ \text{ for } \ i \neq j.$$

We indicate also by $\mathcal{N}^i_K$ and $\mathcal{N}_K$ the restriction of $\mathcal{N}^i_{KL}$ and $\mathcal{N}_{KL}$ to the single coarse cell $K$, respectively. We indicate by $\mathcal{E}_K$ the set of edges of the mesh $\mathcal{M}_K$. Finally, given a fracture edge $\sigma$, we denote by $\mathcal{M}_\sigma$ the set of fracture cells having $\sigma$ as an edge.

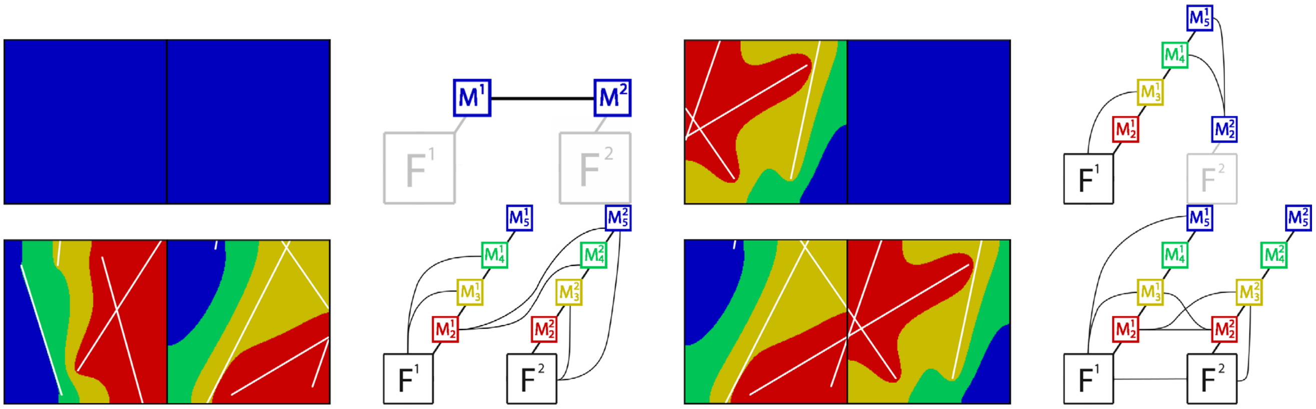

**Figure 2.** Examples of the four possible configurations of pair of coarse cells in which we consider the matrix-matrix connection.

#### 4.1.1. Intercell Upscaled Transmissibility

To compute the upscaled transmissibility between two coarse blocks $K, L \in \mathcal{G}$, which represents the link between the coarse degrees of freedom associated with the local fractures sets of the two cells, we solve a diffusion problem where a pressure gradient is imposed across the boundary of the cell. In practice, we solve the system of equations (2), by the virtual fracture cells approach, with a pressure gradient along the direction of the two coarse cells and a no-flux boundary condition over the remaining boundaries. We employ a ghost-cell approach for both fracture and matrix cells to impose the boundary conditions [see *LeVeque*, 2004]. The upscaling procedure is based on the formula (3), where averaged and global quantities are considered. Once the numerical solution is computed, it is possible to calculate the average pressure on each coarse cell and the flux across the coarse cells interface. The number of local networks of fractures which actually cross the interface $\Gamma_{KL}$ is indicated by $N^{\Gamma}_{KL}$, and for a given $j=1,\ldots,N^{\Gamma}_{KL}$, we denote with $\mathcal{N}^j_{KL}=\mathcal{N}^j_K \cup \mathcal{N}^j_L$ the *j*th local network contained in the pair of cells *K* and *L*. For each local network of fractures of the coarse cell *K*, the averaged pressure $\bar{p}^j_K$ is computed by

$$\bar{p}^j_K = \frac{\sum_{n\in\mathcal{N}^j_K} |n| p_n}{\sum_{n\in\mathcal{N}^j_K} |n|} \quad \text{for } j=1,\ldots,N^{\Gamma}_{KL},$$

where $n$ is a fracture cell of the mesh associated with $\mathcal{N}^j_K$. The same formula is applied analogously for *L*. It is important to remark that only the fracture cells are involved in the computation of $\bar{p}^j_K$ and $\bar{p}^j_L$. The total flux, indicated by $F^j_{KL}$, through each fractures network $\mathcal{N}^j_{KL}$ across the interface between the two cells is computed by

$$F^j_{KL}(p) = \sum_{\sigma\in\mathcal{E}^j_\Gamma} T_{nm}(p_n - p_m) \quad \text{for } j=1,\ldots,N^{\Gamma}_{KL},$$

where $\mathcal{E}^j_\Gamma := \mathcal{E}_{\mathcal{N}^j_K} \cap \mathcal{E}_{\mathcal{N}^j_L}$ is the set of the interface edges and $n$ ($m$) represent the fracture cell of the mesh associated with $\mathcal{N}^j_K$ ($\mathcal{N}^j_L$) such that $n$ ($m$) $\in \mathcal{M}_\sigma$. Finally, the upscaled transmissibility between *K* and *L* can be computed considering

$$T_{KL} = \mu \sum_{j=1}^{N^{\Gamma}_{KL}} \left| \frac{F^j_{KL}(p)}{\bar{p}^j_K - \bar{p}^j_L} \right|. \tag{7}$$

Referring only to the fracture boxes in *Karimi-Fard et al.* [2006, Figure 2], each node represents the system of fracture inside the coarse cell, while each link between the nodes represents the connection, i.e., the transmissibility.

*Remark 2.* Conceptually, formula (7) describes more accurately the flux exchange across the interface since all the fractures entering the interface are treated like conductances in parallel.

#### 4.1.2. Intracell Upscaled Transmissibility

Once the upscaled transmissibility is computed for each pair of cells in the coarse grid, the multiple subregion approach subdivides each single coarse cell $K \in \mathcal{G}$ into several parts and computes the upscaled transmissibility between the degrees of freedom associated with the fractures set of the cell and the first subregion, as well as the transmissibilities between a subregion and the subsequent one. To split the coarse cell, *Karimi-Fard et al.* [2006] propose considering the compressible single-phase flow equation for the generic coarse cell $K \in \mathcal{G}$

$$\begin{cases} \phi c \dfrac{\partial p}{\partial t} - \nabla \cdot \dfrac{\Lambda}{\mu} \nabla p = q_f & \text{in } K \text{ for } t > 0 \\ -\dfrac{\Lambda}{\mu} \nabla p \cdot \boldsymbol{n}_K = 0 & \text{on } \partial K \text{ for } t > 0 \\ p = 0 & \text{in } K \text{ for } t = 0 \end{cases}, \tag{8}$$

where $\phi$ is the porosity, $c$ ($Pa^{-1}$) the compressibility, and $q_f$ is a piecewise constant source term strictly positive inside the fractures and zero elsewhere. This equation is solved until a pseudo steady state is reached, i.e., until the following condition is fulfilled:

$$\frac{\partial p}{\partial t}=c_1=const.$$

Problem (8) is weakly coercive and the numerical errors accumulated during the temporal loop may alter the computed solution. Finding the pseudo steady state solution of problem (8) can be proved to be equivalent to finding the solution of the following stationary problem:

$$\begin{cases} -\nabla\cdot\frac{\Lambda}{\mu}\nabla p=q_f-\frac{\phi}{\int_K\phi}\int_K q_f & \text{in } K \\ -\frac{\Lambda}{\mu}\nabla p\cdot \boldsymbol{n}_K=0 & \text{on } \partial K \\ \int_K p=0 \end{cases}. \tag{9}$$

Since only homogeneous Neumann boundary conditions are considered for the previous system, the last condition is needed to ensure the uniqueness of the pressure. To reduce the computational time, in all the tests performed, we will consider only the stationary formulation (9) of problem (8). To numerically solve problem (9), we consider again a fine mesh $\mathcal{M}_K$ built from $K$, and we use the two-point flux approximation to compute the numerical solution. We consider $S_K$ subregions for the coarse cell $K\in\mathcal{G}$. The last subregion is related to the fractures' set. The other $S_K-1$ are defined through the iso-pressure contours on the fine grid $\mathcal{M}_K$. Practically, we divide the pressure range, only for the matrix, uniformly in $S_K-1$ parts. Thanks to the characteristics of problem (8), each subregion is linked between the previous and the subsequent one. Once the subregions are defined, it is possible to compute the transmissibility between each pair of subregions considering a proper generalization of formula (3). We indicate by $\mathcal{M}_K^i$ the cells of the fine mesh $\mathcal{M}_K$ related to the subregion $i$ and by $\mathcal{F}_K^i$ the facets related to $\mathcal{M}_K^i$. We consider also the set of facets between two subregions as $\mathcal{F}_K^{i,i+1}=\mathcal{F}_K^i\cap\mathcal{F}_K^{i+1}$. Finally, we set $\mathcal{M}_K^{S_K}=\mathcal{F}_K^{S_K-1,S_K}$ the mesh related to $\mathcal{N}_K$. We compute the mean pressure for each subregion $i$ as

$$\bar{p}_K^i=\frac{\sum_{k\in\mathcal{M}_K^i}|k|p_k}{\sum_{k\in\mathcal{M}_K^i}|k|} \quad \text{for } i=1,\ldots,S_K,$$

as well as the flux between the subregions $i$ and $i+1$ as

$$F_K^{i,i+1}(p)=\sum_{f\in\mathcal{F}_K^{i,i+1}} T_{kl}(p_k-p_l) \quad \text{for } i=1,\ldots,S_K-1,$$

where $k\in\mathcal{M}_K^i$ and $l\in\mathcal{M}_K^{i+1}$ are two fine cells. The upscaled transmissibility between the subregions $i$ and $i+1$ is computed by

$$T_K^{i,i+1}=\mu\left|\frac{F_K^{i,i+1}(p)}{\bar{p}_K^i-\bar{p}_K^{i+1}}\right|. \tag{10}$$

Considering only one fracture branch in *Karimi-Fard et al.* [2006, Figure 2], it presents the case of a single coarse cell with the fractures and the subregions. The link between the node related to the fractures and the first subregion represents the fracture-matrix transmissibility, while the remaining links represent the transmissibility between subsequent subregions.

*Remark 3.* The fractures that are fully immersed, i.e., the fractures that are contained only in one coarse cell, are considered as part of the subregion to which they belong (see the immersed fracture contained in the left coarse cell of the bottom left of Figure 2 which belongs to the green matrix subregion). In this case the source term, in (8), associated with these fractures is set to zero.

*Remark 4.* The assumption of logically one-dimensional connectivity, made in *Gong et al.* [2006, 2008b] and *Tatomir et al.* [2011], does not hold any more. We allow that each subregion can be linked to all the other subregions. This is particularly important when the local mesh of each coarse cell contains few cells.

### 4.2. Matrix-Matrix Communication

In this section, following the idea presented in *Hui et al*. [2007], we introduce an extension of the previous algorithm to deal with disconnected networks of fractures in the upscaling procedure. Disconnected networks of fractures occur when there are pairs of coarse cells where the local set of fractures does not intersect the interface between the coarse cells. In this case, the previous algorithm does not compute any upscaled transmissibility since one of the hypotheses of the standard multiple subregions method is that the flow, between adjacent coarse cells, takes place only through the fractures. Figure 2 shows this kind of configurations. In particular, we subdivide the cells into three cases: both cells do not contain any fracture; only one cell contains the fractures; both cells contain fractures but they do not cut the common interface. Besides these cases, for a more accurate representation, we extend the procedure described below also to the case in which the common interface is cut by the fractures.

To handle these cases, we take advantage of the created subregions during the computation of the upscaled transmissibilities inside each coarse cell. Given a pair of adjacent coarse cells, where their networks of fractures are disconnected, we solve the diffusion problem (2), where a pressure gradient is imposed across their boundary. Once the solution is obtained, we compute the transmissibilities between each subregion of the first coarse cell which faces, through the interface between the coarse cells, any subregions of the other coarse cell. An example is given in Figure 2 (bottom left), where the subregions of a pair of coarse cells are represented. The graph depicts the logical connections between the d.o.f. associated with the subregions: the subregions of the left cell (indicated with 1) are linked to the subregions of the right cell (indicated with 2). For each link we have a transmissibility value. Considering the two fine meshes $\mathcal{M}_K$ and $\mathcal{M}_L$ associated with the pair of coarse cells $K$ and $L \in \mathcal{G}$, it is possible to define the subregions as explained in the previous section obtaining $S_K$ and $S_L$ subregions for the two coarse cells $K$ and $L$, respectively. As in the previous section, the upscaled transmissibility is based on a proper generalization of equation (3). We compute the mean pressure for each subregion $i$ of the coarse cell $K$ as

$$\bar{p}_K^i = \frac{\sum_{k \in \mathcal{M}_K^i} |k| p_k}{\sum_{k \in \mathcal{M}_K^i} |k|},$$

and analogously for the coarse cell $L$. We indicate the set of facets of the subregion $i$ of the coarse cell $K$ which faces the subregion $j$ of the coarse cell $L$ as $\mathcal{F}_{KL}^{i,j}$. The total flux, indicated by $F_{KL}^{i,j}$, through the interface between the cells of the subregion $i$, referred to the coarse cell $K$, and the subregion $j$, referred to the coarse cell $L$, is computed by

$$F_{KL}^{i,j}(p) = \sum_{f \in \mathcal{F}_{KL}^{i,j}} T_{kl}(p_k - p_l),$$

where every fine cell $k \in \mathcal{M}_K^i$ belongs to the subregion $i$ of the coarse cell $K$, and every fine cell $l \in \mathcal{M}_L^j$ belongs to the subregion $j$ of the coarse cell $L$. Finally, the upscaled transmissibility between the subregion $i$ of the coarse cell $K$ and the subregion $j$ of the coarse cell $L$ can be computed considering

$$T_{KL}^{i,j} = \mu \left| \frac{F_{KL}^{i,j}(p)}{\bar{p}_K^i - \bar{p}_L^j} \right|.$$

It is important to notice that the procedure does not increase the number of degrees of freedom but only their connections. In the particular case when one of the two coarse cells does not contain fractures, as represented in Figure 2 (top right), we assume that the latter has only one subregion and the upscaling algorithm remains the same. All the subregions facing the interface between the coarse cells of the left coarse cell are connected with the unique subregion of the right coarse cell. This procedure is particularly important in the "boundary" of a fractures network to allow the spreading of the flow in the neighboring coarse cells, and to link different networks of fractures.

To complete the upscaling procedure for a general reservoir, it is possible that some pairs of coarse cells do not contain any fracture. In this case, we consider the standard formula (4) and (5), with the geometric properties related to the coarse mesh $\mathcal{G}$. This case is represented by the graph shown in Figure 2 (top left). The degrees of freedom related to the fractures are not considered, and for each coarse cell there is a single d.o.f. associated with the unique matrix subregion, i.e., the entire coarse cell. Hence, the link between the coarse cells is due to the flow through the porous matrix.

Finally, to improve the accuracy of the procedure, we compute the matrix-matrix transmissibility also in the case of pair of cells in which the common interface is fractured. Indeed, though most of the flow goes through the fractures, the interface area associated with the porous matrix may be very large with respect to the one related to the fractures, so that the flow transmitted by the matrix may be significant. An example of this case is shown in Figure 2 (bottom right): besides the fracture-fracture transmissibility, a communication between the subregions of the two coarse cells is taken into account.

*Remark 5*. With the basic algorithm presented in *Gong et al*. [2006] and *Tatomir et al*. [2011], it is possible to locally eliminate the matrix d.o.f. in favor of the fracture d.o.f., reducing the upscaled system. It is worth noticing that considering matrix-matrix communication among adjacent coarse cells prevents this possibility to occur, and the number of unknowns in the coarse system may increase considerably.

*Remark 6*. To speed up the resolution in the implementation of the algorithm, it is possible to avoid the computation of the transmissibilities among the subregions if the local set of fractures belongs to the same global network of fractures. This hypotheses is reasonable since the transmissibility of the fractures is several orders of magnitude greater than the transmissibility of the rock matrix.

The whole procedure presented in this section is summarized in Algorithm 1.

**Algorithm 1.** Upscaling procedure

```
Require: coarse grid, fractures
Ensure: transmissibility, porous volume, depth
for all coarse cell do
    compute intersection among the local grid and fractures
    solve problem (9)
    compute the subregions
    compute the porous volume and the depth for each region
    compute intracell upscaled transmissibility
end for
for all pair of coarse cells do
    compute intersections among the local grid and fractures
    solve the problem (2) by enforcing a pressure jump
    compute intercell upscaled transmissibility for both the fracture and the matrix
end for
```

## 5. Numerical Examples

In this section, we present several numerical examples to assess the performance and the validity of the method presented in the previous sections. In section 5.1, we compare the computational cost of the EDFM procedure with respect to the constrained Delaunay algorithm. We then proceed in section 5.2 to show the convergence of the upscaled transmissibilities computed with EDFM with respect to the grid refinement. In section 5.3, we compare the numerical solution obtained with Intersect, that uses Oda's method to compute the transmissibilities, the one obtained with Eclipse and by computing the upscaled transmissibilities with the commercial software Petrel, and finally the numerical solution obtained with Eclipse but by computing the upscaled transmissibilities with our upscaling procedure. In section 5.4, we assess the upscaling procedure in a complex 3-D framework and, in particular, we compare the numerical solution obtained with GPRS by considering a different number of subregions during the upscaling procedure. In sections 5.3 and 5.4, we refer to the two-phase problem described in *Bastian* [1999, p. 27, equation (1.41)].

### 5.1. Computational Efficiency of EDFM

We present a comparison of the computational cost between EDFM, introduced in section 3, and a classical constrained Delaunay approach. We consider two test cases: the first is a set of 64 fractures (Figure 3, top), the second a set of 1024 fractures (Figure 3, bottom). For the generation of the Dealunay meshes, the popular TetGen program [*Si*, 2015] is employed. This generates tetrahedral meshes of arbitrary 3-D polyhedral domains using exact constrained Delaunay tetrahedralizations, boundary conforming Delaunay meshes, and Voronoi partitions. The resulting Delaunay meshes consists of 92,367 rock matrix cells for the test case with 64 fractures, and 1,363,190 cells for the test case with 1024 fractures. For each test case, EDFM was run using two different Cartesian grids. It is worth to notice that we do not exploit the fact that the meshes are Cartesian but we use the algorithms for general corner-point grids. The first was chosen so that it matched approximately the number of elements obtained using the Delaunay approach (80×80×16 cells for the 64 fractures case and 200×200×200 for the 1024 fractures case). The second was instead chosen to match the constraint on the cell size imposed by TetGen (20×20×4 cells for the 64 fractures case and 50×50×10 for the 1024 fractures case).

*Remark 7*. While the comparison between a Delaunay approach and EDFM using roughly the same number of cells might provide results about the efficiency of our implementation of EDFM, the comparison between meshes with the same cell dimensions is the one proving the usefulness of EDFM: while Delaunay needs to refine the mesh in the proximity of the fractures, using EDFM avoids the need for any refinement of the rock matrix mesh.

For the test case with 64 fractures, TetGen runs in 0.44*s*, while EDFM takes 0.82*s* for the 80×80×16 corner-point grid, and 0.04*s* for the 20×20×4 grid. When run on the test case with 1024 fractures, instead, TetGen takes 16.94*s* and EDFM takes 10.26*s* for the 200×200×200 grid, and 0.45*s* for the 20×20 ×4 grid.

Even when the comparison was made using the same number of cells, EDFM still performs better than a constrained Delaunay approach, even though this is generally not a realistic usage of EDFM, as one of its main advantages is indeed the fact that it does not use a conforming mesh, eliminating the need for such a refinement. The fact that EDFM performs better with a greater number of fractures, moreover, is a sign of its better scalability when compared to a conforming meshing strategy such as a Delaunay tetrahedralizations.

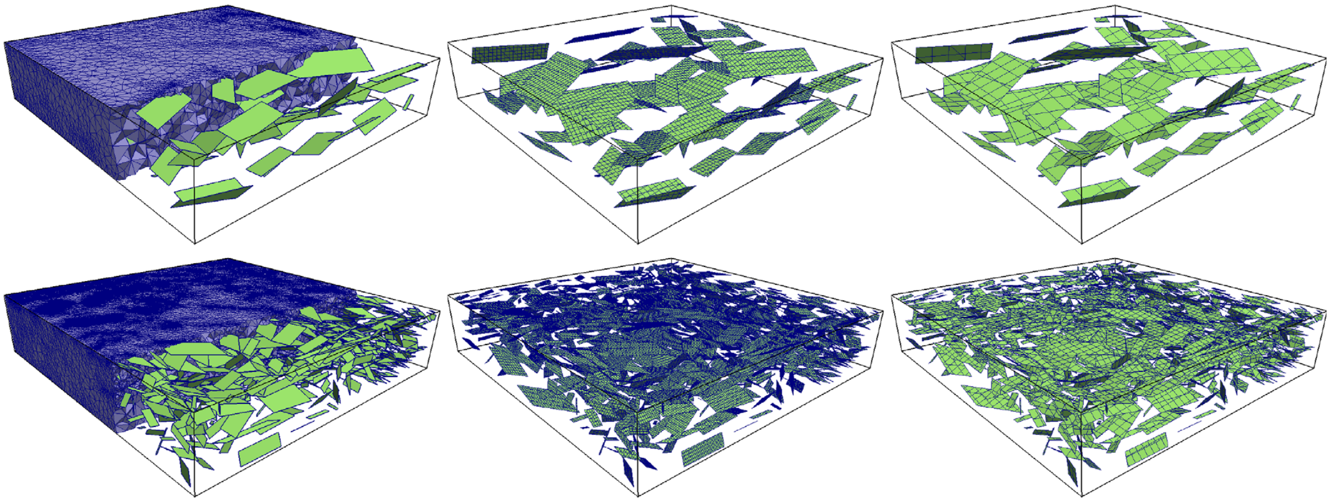

**Figure 3.** (top) Test case with 64 fractures, meshed using a constrained Delaunay approach (left), EDFM with approximately the same number of elements (middle) and EDFM with the same maximum element volume imposed with the Delaunay triangulation. (right) bottom: test case with 64 fractures, meshed using a constrained Delaunay approach (left), EDFM with approximately the same number of elements (middle) and EDFM with the same maximum element volume imposed with the Delaunay triangulation (right).

*Remark 8.* Both fracture sets where generated so that no intersections between fractures occurs. This was necessary as TetGen is not capable of handling intersecting faces, nor did we have any such tool at our disposal. Note, however, that should the fractures intersect each other, this would not affect significantly EDFM, while it could prove a major obstacle for a Delaunay mesh builder.

### 5.2. Convergence of the Upscaled Fracture-Matrix Transmissibility Computed With EDFM

In order to ascertain the validity of our approach (i.e., the use of EDFM to compute upscaled transmissibilities), we consider a test case with a 2×2 grid and four fractures (see Figure 4, top). The domain is a cube of edge length 1 m and the fractures have an aperture of 0.1 m. The matrix permeability is $10^{-12}$ m$^2$ while the fracture permeability is $10^{-8}$m$^2$. For each coarse cell, we evaluate the transmissibility between the degree of freedom associated with the set of fractures of the cell and the first subregion, see (10), by applying EDFM with different local grid refinements. In particular, for each coarse cell, the refined grid is generated as a Cartesian grid with $n \times n$ elements, for $n=5, 25, 50, 75, 100$. This computation is done for a number of matrix subregions equal to 1, 3, and 5. It can be seen from Figure 4 that for each of the four coarse cells and for each of the three number of subregions tested, the computed transmissibility converges toward a single value.

*Remark 9.* Transmissibilities computed for different number of subregions are expected to be different from one another as the shape of the first matrix subregion differs depending on the selected number of subregions.

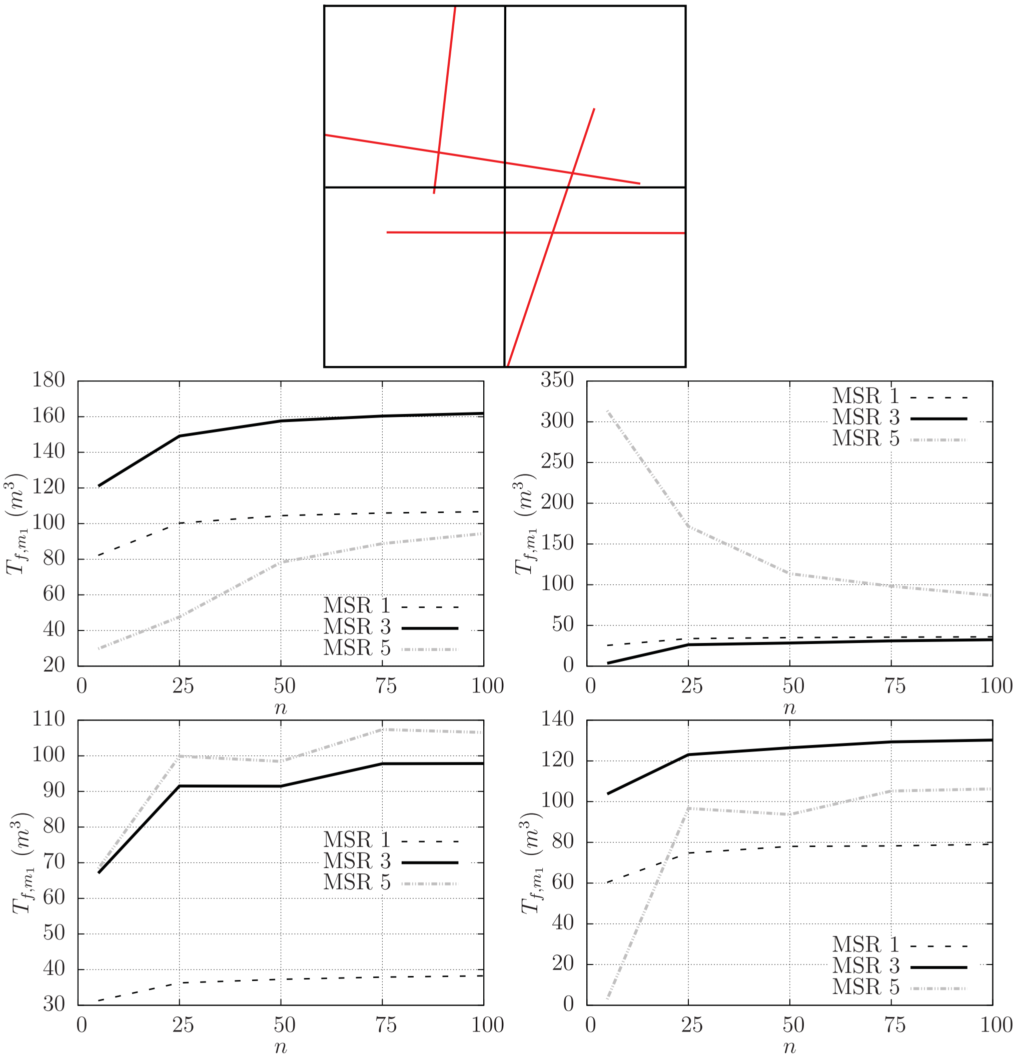

**Figure 4.** 2 × 2 test case (top) used to show the convergence of the upscaled transmissibility between the fractures of a coarse cell and the first subregion. The four graphs show the computed transmissibility, respectively, for the top left, top right, bottom left, and bottom right cell.

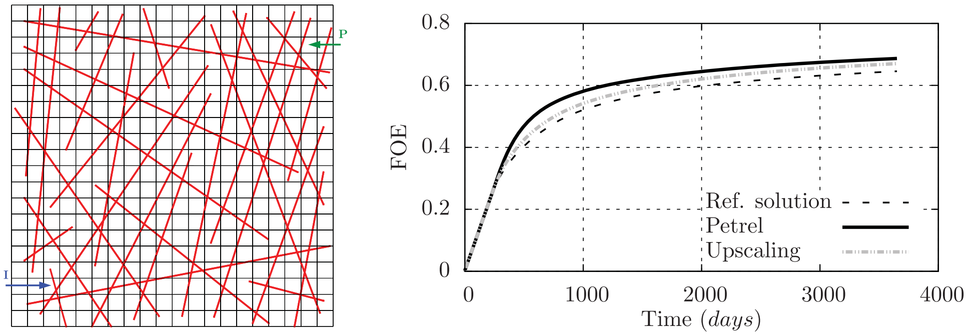

**Figure 5.** Left: coarse mesh and fractures for the comparison between the upscaling procedure and Petrel. Right: comparison of the field oil efficiency for the upscaling, Petrel and the reference solution.

### 5.3. Comparison With a Standard Industrial Simulator

In this section, we compare the result obtained with the upscaling procedure [*Petrel*, 2014], considered as a standard simulation tool in oil industry, against a reference solution. The upscaling considers only one porous matrix subregion since, as shown in the previous part, increasing the number of subregions, we expect a better representation of the coarse solution. The test domain is represented in Figure 5 (left), where the fractures and the 20 × 20 coarse mesh are represented. The fracture permeability is 5 orders of magnitude greater than the matrix permeability and the fracture thickness is on the order of centimeters. The reference solution is computed using EDFM with 100 × 100 Cartesian cells. Petrel uses the geometry-based Oda's method [*Oda*, 1985], to compute a representative permeability tensors initially defined in the discrete fracture network. The Kazemi formula is used to compute the transmissibility between the fractures and the rock matrix [*Kazemi et al*., 1976]. As in the previous test case, we consider an incompressible two-phase flow simulation where the reservoir and the fractures are filled with oil. Water is pumped in at a constant rate in the injection well, *I*, whereas the production well, *P*, is operating at a constant pressure. Both wells are completed in fracture. For the computation of the saturation field, we use Eclipse [*Eclipse*, 2014] for both the upscaling and Petrel solutions and Intersect [*Intersect*, 2014] for the reference solution. The comparison of the field oil efficiency (FOE) for the three approaches is reported in Figure 5 (right). The FOE defined as

$$FOE_t = \frac{OIP_{t=0} - OIP_t}{OIP_{t=0}},$$

where *OIP* is the so-called oil-in-place, which refers to the total oil content of a reservoir. In the beginning of the simulation, the three solutions are comparable. However, after 400 days the solution computed using upscaling approximates better the reference solution than the one computed using Petrel. In Figure 6, the oil saturation profile for the three methods is provided at the same time level. Even though the upscaled solution shows a delay compared to the reference solution, it is able to better recover the oil dynamic than Petrel.

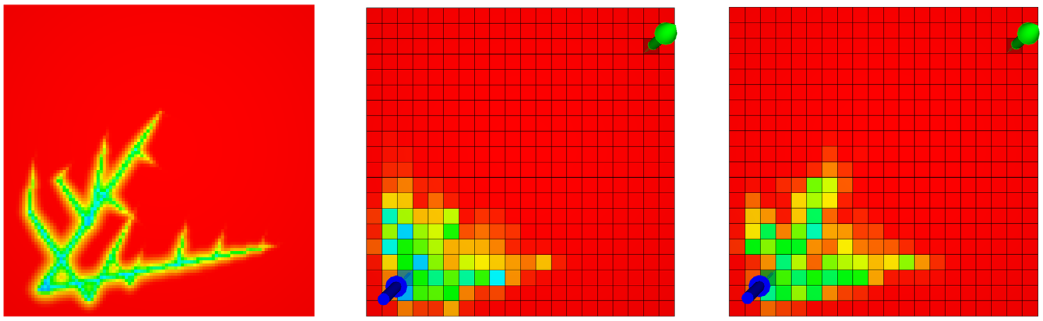

**Figure 6.** Representation of the oil saturation in the porous matrix at the same time level. Reference solution (left), solution obtained using Petrel (middle), and solution obtained using the upscaling (right).

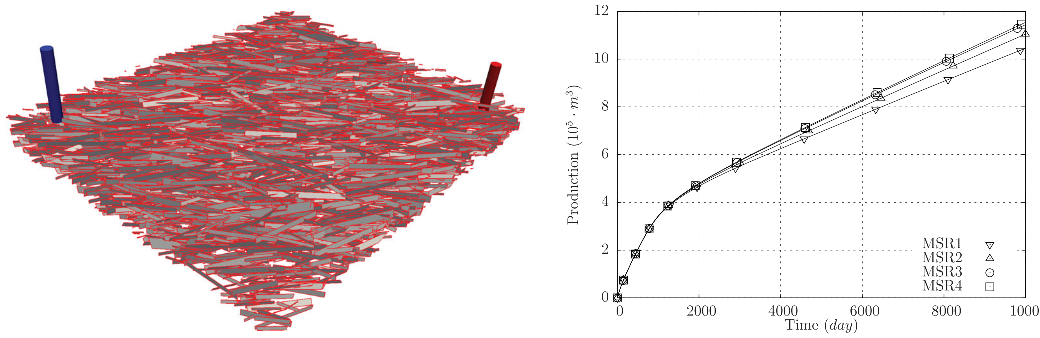

**Figure 7.** (left) Sketch of the fractures and location of the injector well (blue) and production well (red). (right) Oil produced by the production well as a function of time by considering different number of subregions.

### 5.4. Realistic Three-Dimensional Case

The General Purpose Reservoir Simulator (GPRS), first introduced in *Cao* [2002], is a general purpose research simulator developed and maintained by Stanford University's Reservoir Simulation (SUPRI-B) Industrial Affiliates Program. GPRS incorporates all techniques of modern reservoir simulations such as black oil or compositional fluid modeling, employment of several adaptive implicit formulations, management of structured and unstructured grids, two-point and multipoint flux discretizations, and so on.

In this example, the goal is to perform a numerical simulation on a complex three-dimensional set of fractures in order to test the robustness and the efficiency of the procedure. Moreover, we assess the dependence of the numerical solution with respect to the number of subregions. We consider a test case with a realistic fracture distribution where the coarse grid is Cartesian made of 30×25×15 cells which contains a set of 4241 fractures. The domain is about 7 km along its diameter. In the solution of the local problems, each coarse cell is divided into 400 fine cells, the matrix permeability is set equal to $9.868\times10^{-15}\,\text{m}^2$, the matrix porosity equal to 0.25, the fracture permeability is $9.868\times10^{-10}\,\text{m}^2$, the fracture porosity equal to 0.8 and a fracture aperture in between 0.1 and 0.3 m. GPRS is used to simulate a two-phase (oil-water) flow in a fractured porous medium with the upscaled values of transmissibility, porous volumes, and mean depths. In particular, we consider the reservoir saturated by oil, with density equal to $786\ \text{kg}/\text{m}^3$, in which water is pumped in, with density equal to $1038\ \text{kg}/\text{m}^3$. The injector well operates at a fixed flow rate equal to $11\,\text{m}^3/\text{s}$, while the producer well works at a fixed bottom-hole pressure equal to 689,476 Pa. The wells are completed in the fractures contained in the coarse cell, with a transmissibility high enough so that the dynamics between the well and the fractures is instantaneous. The rock compressibility is set to $1.45\times10^{-10}\,\text{Pa}^{-1}$. Moreover, we neglect the residual saturation for both phases and the capillary pressure, we assume a simple linear relation for the relative permeability and we apply a hydrostatic pressure as initial pressure condition. We place the water injector well in the upper part on the left corner of the grid and the production well in the lower part on the right corner of the grid, referring to Figure 7 (left). In this test, we perform the upscaling procedure four times by changing the number of subregions, from one to four in a dual porosity-dual permeability framework. Figure 7 (right) shows the amount of oil produced from the production well with different number of subregions. We note that by increasing the number of subregions, the oil produced increases. Regarding the computational effort, the upscaling procedure requires 2176*s*, 2157*s*, 2157*s*, and 2158*s* for the case of 1, 2, 3, and 4 subregions, respectively. In particular, 10 s are required to solve all the single cell problems and 48 s to solve all the pair cell problems. The computational time is independent of the number of subregions. Instead, the computational time employed to solve the two-phase flow simulation with GPRS is 378*s*, 724*s*, 1236*s*, and 1903*s* for the case of 1, 2, 3, and 4 subregions, respectively.

## 6. Conclusions

In this work, we presented a novel efficient upscaling procedure to derive a coarse-scale model for a natural fractured reservoir. The main objective of the proposed scheme is to compute the upscaled parameters in an efficient and effective way. These parameters are computed via local problems which involve single or pair of coarse cells. To achieve a good efficiency, we have considered, in the solution of these problems, a nonconforming coupling between the bulk grid and the fracture grids by means of the embedded discrete fracture model method. Moreover, we considered an extension of the standard MSR algorithm considering the full connection of the subregions among facing cells. As the numerical tests show, the upscaled problem turns out to be a reasonable and correct approximation for this class of problems. Moreover, the methodology introduced yields better results than the standard software used in oil industry to solve similar problems. In a forthcoming work, we will introduce a parallelization of the global algorithm to increase the efficiency and applicability to real problems.

## Appendix A: EDFM Intersection Algorithm

We describe here the algorithm used to compute the intersections between a bilinear surface and a segment. Given the parameterization of a bilinear surface of vertices ***a***, ***b***, ***c***, and ***d***

$$\boldsymbol{r}(u,v)=v[u\boldsymbol{b}+(1-u)\boldsymbol{a}]+(1-v)[u\boldsymbol{c}+(1-u)\boldsymbol{d}],$$

finding the intersections with a line segment

$$\boldsymbol{s}(t)=t\boldsymbol{p}+(1-t)\boldsymbol{q},$$

with endpoints ***p*** and ***q***, entails solving the nonlinear system

$$\boldsymbol{r}(u,v)=\boldsymbol{s}(t). \tag{A1}$$

In order to solve (A1), we need to know how to solve two subproblems.

### A1. Bilinear System of Equations

The first is finding the solution to the following nonlinear system of equations for the variables *u* and *v*:

$$\begin{cases} \alpha_0 u+\beta_0 v+\gamma_0 uv+\delta_0=0 \\ \alpha_1 u+\beta_1 v+\gamma_1 uv+\delta_1=0 \end{cases}, \tag{A2}$$

which we will refer to as a bilinear system. Despite being nonlinear, this type of system can be solved explicitly using the decision tree depicted in Figure 8. Particular care has to be taken to deal with numerical errors and floating-point arithmetic.

### A2. Parameterization of a Point

The second subproblem is finding the parameterization of a point ***x*** with respect to a given bilinear surface $\boldsymbol{r}(u,v)$. This means solving $\boldsymbol{r}(u,v)=\boldsymbol{x}$, which is a system of three equations in two variables (*u* and *v*)

$$\begin{cases} (b_x-a_x)u+(d_x-a_x)v+(a_x-b_x+c_x-d_x)uv+a=x_x \\ (b_y-a_y)u+(d_y-a_y)v+(a_y-b_y+c_y-d_y)uy+a=x_y \\ (b_z-a_z)u+(d_z-a_z)v+(a_z-b_z+c_z-d_z)uz+a=x_z \end{cases}. \tag{A3}$$

This is solved by considering each possible pair of equations, finding their solutions, and then checking whether the third equation holds (to within a given tolerance) for the solutions found. Each pair of equations is a bilinear system that can be solved as described in Figure 8. Usually, we are only interested in solutions lying inside the unit square $[0,1]^2$, as these are the ones inside the fracture or face.

### A3. Solving the Nonlinear System of Equations

Equation (A1) is a system of three equations in three variables (*u*, *v*, and *t*)

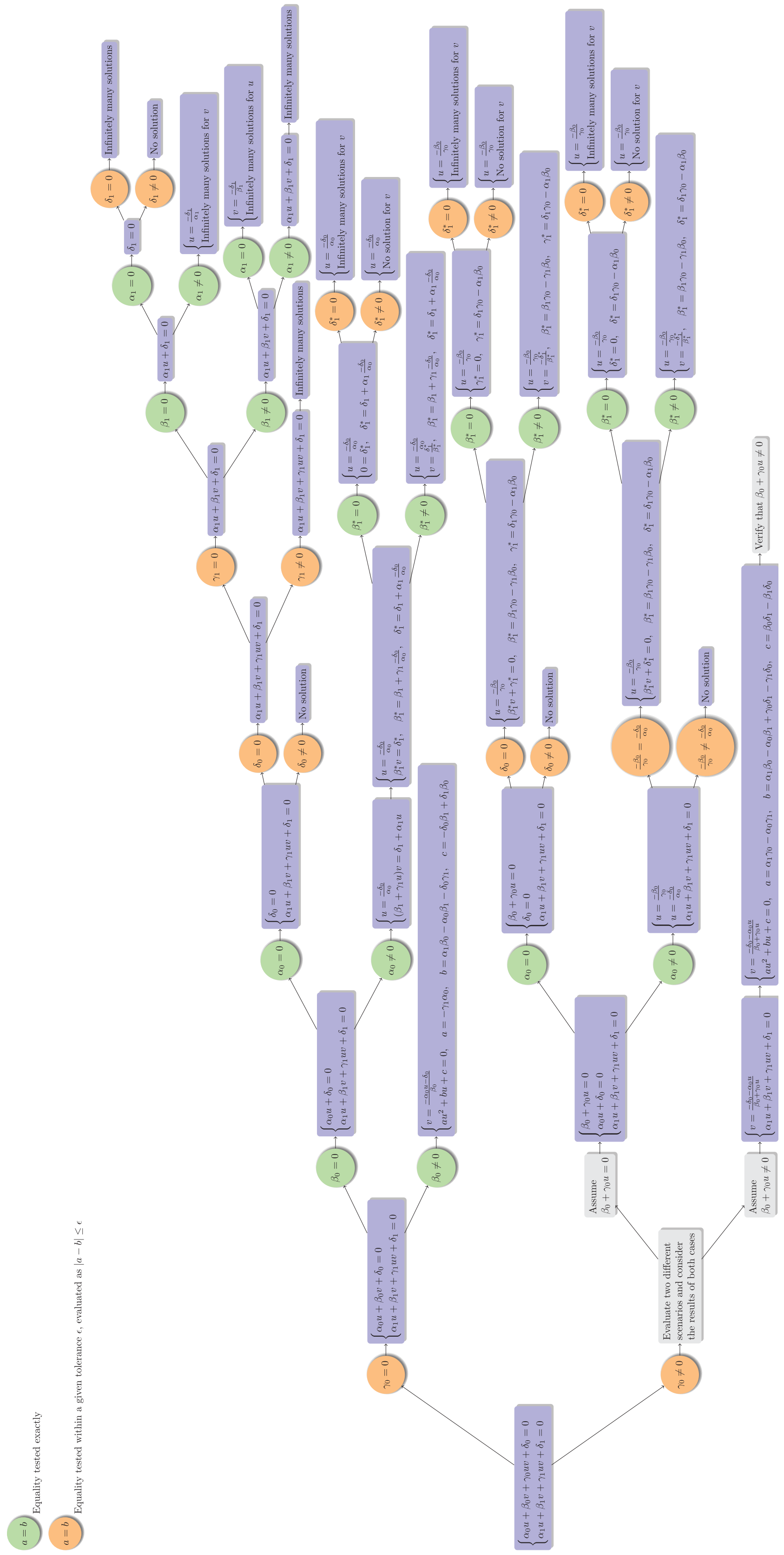
$a = b$ Equality tested exactly
$a = b$ Equality tested within a given tolerance $\epsilon$, evaluated as $|a - b| \leq \epsilon$
$\begin{cases} \alpha_0 u + \beta_0 v + \gamma_0 uv + \delta_0 = 0 \\ \alpha_1 u + \beta_1 v + \gamma_1 uv + \delta_1 = 0 \end{cases}$
$\gamma_0 = 0$
$\gamma_0 \neq 0$
$\begin{cases} \alpha_0 u + \beta_0 v + \delta_0 = 0 \\ \alpha_1 u + \beta_1 v + \gamma_1 uv + \delta_1 = 0 \end{cases}$
$\beta_0 = 0$
$\beta_0 \neq 0$
$\begin{cases} \alpha_0 u + \delta_0 = 0 \\ \alpha_1 u + \beta_1 v + \gamma_1 uv + \delta_1 = 0 \end{cases}$
$\alpha_0 = 0$
$\alpha_0 \neq 0$
$\begin{cases} \delta_0 = 0 \\ \alpha_1 u + \beta_1 v + \gamma_1 uv + \delta_1 = 0 \end{cases}$
$\delta_0 = 0$
$\delta_0 \neq 0$
No solution
$\alpha_1 u + \beta_1 v + \gamma_1 uv + \delta_1 = 0$
$\gamma_1 = 0$
$\gamma_1 \neq 0$
$\alpha_1 u + \beta_1 v + \delta_1 = 0$
$\alpha_1 u + \beta_1 v + \gamma_1 uv + \delta_1 = 0$
Infinitely many solutions
$\beta_1 = 0$
$\beta_1 \neq 0$
$\alpha_1 u + \delta_1 = 0$
$\alpha_1 u + \beta_1 v + \delta_1 = 0$
$\alpha_1 = 0$
$\alpha_1 \neq 0$
$\delta_1 = 0$
$\delta_1 = 0$
$\delta_1 \neq 0$
Infinitely many solutions
No solution
$\begin{cases} u = \frac{-\delta_1}{\alpha_1} \\ \text{Infinitely many solutions for } v \end{cases}$
$\alpha_1 = 0$
$\alpha_1 \neq 0$
$\begin{cases} v = \frac{-\delta_1}{\beta_1} \\ \text{Infinitely many solutions for } u \end{cases}$
$\alpha_1 u + \beta_1 v + \delta_1 = 0 \to$ Infinitely many solutions
$\begin{cases} u = \frac{-\delta_0}{\alpha_0} \\ (\beta_1 + \gamma_1 u)v = \delta_1 + \alpha_1 u \end{cases}$
$\begin{cases} u = \frac{-\delta_0}{\alpha_0} \\ \beta_1^* v = \delta_1^*, \end{cases} \quad \beta_1^* = \beta_1 + \gamma_1 \frac{-\delta_0}{\alpha_0}, \quad \delta_1^* = \delta_1 + \alpha_1 \frac{-\delta_0}{\alpha_0}$
$\beta_1^* = 0$
$\beta_1^* \neq 0$
$\begin{cases} u = \frac{-\delta_0}{\alpha_0} \\ 0 = \delta_1^*, \end{cases} \quad \delta_1^* = \delta_1 + \alpha_1 \frac{-\delta_0}{\alpha_0}$
$\begin{cases} u = \frac{-\delta_0}{\alpha_0} \\ v = \frac{\delta_1^*}{\beta_1^*}, \end{cases} \quad \beta_1^* = \beta_1 + \gamma_1 \frac{-\delta_0}{\alpha_0}, \quad \delta_1^* = \delta_1 + \alpha_1 \frac{-\delta_0}{\alpha_0}$
$\delta_1^* = 0$
$\delta_1^* \neq 0$
$\begin{cases} u = \frac{-\delta_0}{\alpha_0} \\ \text{Infinitely many solutions for } v \end{cases}$
$\begin{cases} u = \frac{-\delta_0}{\alpha_0} \\ \text{No solution for } v \end{cases}$
$\begin{cases} v = \frac{-\alpha_0 u - \delta_0}{\beta_0} \\ au^2 + bu + c = 0, \end{cases} \quad a = -\gamma_1 \alpha_0, \quad b = \alpha_1 \beta_0 - \alpha_0 \beta_1 - \delta_0 \gamma_1, \quad c = -\delta_0 \beta_1 + \delta_1 \beta_0$
Evaluate two different scenarios and consider the results of both cases
Assume $\beta_0 + \gamma_0 u = 0$
Assume $\beta_0 + \gamma_0 u \neq 0$
$\begin{cases} \beta_0 + \gamma_0 u = 0 \\ \alpha_0 u + \delta_0 = 0 \\ \alpha_1 u + \beta_1 v + \gamma_1 uv + \delta_1 = 0 \end{cases}$
$\alpha_0 = 0$
$\alpha_0 \neq 0$
$\begin{cases} \beta_0 + \gamma_0 u = 0 \\ \delta_0 = 0 \\ \alpha_1 u + \beta_1 v + \gamma_1 uv + \delta_1 = 0 \end{cases}$
$\delta_0 = 0$
$\delta_0 \neq 0$
No solution
$\begin{cases} u = \frac{-\beta_0}{\gamma_0} \\ \beta_1^* v + \gamma_1^* = 0, \end{cases} \quad \beta_1^* = \beta_1 \gamma_0 - \gamma_1 \beta_0, \quad \gamma_1^* = \delta_1 \gamma_0 - \alpha_1 \beta_0$
$\beta_1^* = 0$
$\beta_1^* \neq 0$
$\begin{cases} u = \frac{-\beta_0}{\gamma_0} \\ \gamma_1^* = 0, \end{cases} \quad \gamma_1^* = \delta_1 \gamma_0 - \alpha_1 \beta_0$
$\begin{cases} u = \frac{-\beta_0}{\gamma_0} \\ v = \frac{-\gamma_1^*}{\beta_1^*}, \end{cases} \quad \beta_1^* = \beta_1 \gamma_0 - \gamma_1 \beta_0, \quad \gamma_1^* = \delta_1 \gamma_0 - \alpha_1 \beta_0$
$\delta_1^* = 0$
$\delta_1^* \neq 0$
$\begin{cases} u = \frac{-\beta_0}{\gamma_0} \\ \text{Infinitely many solutions for } v \end{cases}$
$\begin{cases} u = \frac{-\beta_0}{\gamma_0} \\ \text{No solution for } v \end{cases}$
$\begin{cases} u = \frac{-\beta_0}{\gamma_0} \\ u = \frac{-\delta_0}{\alpha_0} \\ \alpha_1 u + \beta_1 v + \gamma_1 uv + \delta_1 = 0 \end{cases}$
$\frac{-\beta_0}{\gamma_0} = \frac{-\delta_0}{\alpha_0}$
$\frac{-\beta_0}{\gamma_0} \neq \frac{-\delta_0}{\alpha_0}$
No solution
$\begin{cases} u = \frac{-\beta_0}{\gamma_0} \\ \beta_1^* v + \delta_1^* = 0, \end{cases} \quad \beta_1^* = \beta_1 \gamma_0 - \gamma_1 \beta_0, \quad \delta_1^* = \delta_1 \gamma_0 - \alpha_1 \beta_0$
$\beta_1^* = 0$
$\beta_1^* \neq 0$
$\begin{cases} u = \frac{-\beta_0}{\gamma_0} \\ \delta_1^* = 0, \end{cases} \quad \delta_1^* = \delta_1 \gamma_0 - \alpha_1 \beta_0$
$\begin{cases} u = \frac{-\beta_0}{\gamma_0} \\ v = \frac{-\delta_1^*}{\beta_1^*}, \end{cases} \quad \beta_1^* = \beta_1 \gamma_0 - \gamma_1 \beta_0, \quad \delta_1^* = \delta_1 \gamma_0 - \alpha_1 \beta_0$
$\delta_1^* = 0$
$\delta_1^* \neq 0$
$\begin{cases} u = \frac{-\beta_0}{\gamma_0} \\ \text{Infinitely many solutions for } v \end{cases}$
$\begin{cases} u = \frac{-\beta_0}{\gamma_0} \\ \text{No solution for } v \end{cases}$
$\begin{cases} v = \frac{-\delta_0 - \alpha_0 u}{\beta_0 + \gamma_0 u} \\ \alpha_1 u + \beta_1 v + \gamma_1 uv + \delta_1 = 0 \end{cases}$
$\begin{cases} v = \frac{-\delta_0 - \alpha_0 u}{\beta_0 + \gamma_0 u} \\ au^2 + bu + c = 0, \end{cases} \quad a = \alpha_1 \gamma_0 - \alpha_0 \gamma_1, \quad b = \alpha_1 \beta_0 - \alpha_0 \beta_1 + \gamma_0 \delta_1 - \gamma_1 \delta_0, \quad c = \beta_0 \delta_1 - \beta_1 \delta_0$
Verify that $\beta_0 + \gamma_0 u \neq 0$

**Figure 8.** Decision tree used to solve bilinear systems. Given the eight coefficients $\alpha_0$, $\beta_0$, $\gamma_0$, $\delta_0$, $\alpha_1$, $\beta_1$, $\gamma_1$, and $\delta_1$, one can walk the decision tree from its root to one of the leaves evaluating the condition on the coefficients. Each round node of the tree represents the evaluation of a condition (i.e., an if statement) and the relative subtree is visited only if that condition holds.

$$\begin{cases} (b_x-a_x)u+(d_x-a_x)v+(a_x-b_x+c_x-d_x)uv+a_x=p_x+(q_x-p_x)t \\ (b_y-a_y)u+(d_y-a_y)v+(a_y-b_y+c_y-d_y)uy+a_y=p_y+(q_y-p_y)t \, . \\ (b_z-a_z)u+(d_z-a_z)v+(a_z-b_z+c_z-d_z)uz+a_z=p_z+(q_z-p_z)t \end{cases} \tag{A4}$$

Here the three subscripts $x$, $y$, and $z$ denote the three components of a point. We assume that at least one of the three coefficients of $t$ is non-null, otherwise the segment is degenerate and we can simply solve (A3) to check whether the point belongs to the surface. We can therefore solve one equation for $t$, and substitute what we obtain into the other two equations. If, for instance, $q_z-p_z \neq 0$, then we have

$$\begin{cases} ((b_x-a_x)u+(d_x-a_x)v+(a_x-b_x+c_x-d_x)uv+a)(q_z-p_z)=p_x+(q_x-p_x)((b_z-a_z)u+(d_z-a_z)v+(a_z-b_z+c_z-d_z)uz+a) \\ ((b_y-a_y)u+(d_y-a_y)v+(a_y-b_y+c_y-d_y)uy+a)(q_z-p_z)=p_y+(q_y-p_y)((b_z-a_z)u+(d_z-a_z)v+(a_z-b_z+c_z-d_z)uz+a) \end{cases} .$$

What we obtain is a bilinear system which can once again be solved as in Figure 8. If infinitely many solutions are found, then the segment lies on the bilinear surface, in which case we are only interested in finding the portion of the segment lying inside the four vertices. The endpoints of this intersection are either given by the intersection of the segment with the edges of the bilinear surface, or by the endpoints of the segment itself if they are inside the four vertices of the bilinear surface. This last check can be performed by finding the parametric coordinates of the endpoints by solving (A3).

**Acknowledgments**
The authors wish to thank Alberto Cominelli, Roberto Colin, Luca Formaggia, Matteo Longoni, Paola Panfili, Anna Scotti, and Luca Turconi for many fruitful discussions. The General Purpose Research Simulator (GPRS) developed by the Reservoir Simulation Research Group (SUPRI-B) at Stanford University was used in this work. The authors wish to thank the anonymous reviewers for their valuable comments and suggestions which allowed them to improve the quality of the paper. The geometrical data of sections 5.1 and 5.3 can be found in Supporting Information associated with this publication.

## References

Angot, P., F. Boyer, and F. Hubert (2009), Asymptotic and numerical modelling of flows in fractured porous media, *M2AN Math. Model. Numer. Anal.*, *43*(2), 239–275, doi:10.1051/m2an/2008052.

Bastian, P. (1999), Numerical computation of multiphase flows in porous media, PhD thesis, Univ. of Heidelberg, Heidelberg, Germany. [Available at http://conan.iwr.uni-heidelberg.de/people/peter/pdf/Bastian_habilitationthesis.pdf.]

Bear, J. (1972), *Dynamics of Fluids in Porous Media*, American Elsevier, N. Y. [Available at http://store.doverpublications.com/0486656756.html.]

Brezzi, F., and M. Fortin (1991), *Mixed and Hybrid Finite Element Methods, Comput. Math*., vol. 15, Springer, Berlin.

Cao, H. (2002), Development of techniques for general purpose simulators, PhD thesis, Stanford Univ., Stanford, Calif.

D'Angelo, C., and A. Scotti (2012), A mixed finite element method for Darcy flow in fractured porous media with non-matching grids, *Math. Modell. Numer. Anal.*, *46*(02), 465–489, doi:10.1051/m2an/2011148.

de Araujo Cavalcante Filho, J. S., M. Shakiba, A. Moinfar, and K. Sepehrnoori (2015), Implementation of a preprocessor for embedded discrete fracture modeling in an IMPEC compositional reservoir simulator, in *SPE Reservoir Simulation Symposium*, Soc. of Pet. Eng., 23-25 February, Houston, Tex., doi:10.2118/173289-MS. [Available at https://www.onepetro.org/conference-paper/SPE-173289-MS.]

Durlofsky, L. J. (2003), Upscaling of geocellular models for reservoir flow simulation: A review of recent progress, in *7th International Forum on Reservoir Simulation*, pp. 23–27. [Available at http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.122.5524.]

Eclipse (2014), Industry reference reservoir simulator. [Available at www.software.slb.com/products/foundation/pages/eclipse.aspx.]

Eymard, R., T. Gallouët, and R. Herbin (2000), Finite volume methods, in Solution of Equation in $\mathcal{R}^n$ *(Part 3), Techniques of Scientific Computing (Part 3), Handbook of Numerical Analysis*, vol. 7, edited by P. G. Ciarlet and J. L. Lions, pp. 713–1018, Elsevier, doi:10.1016/S1570-8659(00)07005-8. [Available at http://www.sciencedirect.com/science/article/pii/S1570865900070058.]

Faille, I., A. Fumagalli, J. Jaffré, and J. E. Roberts (2016), Model reduction and discretization using hybrid finite volumes of flow in porous media containing faults, *Comput. Geosci.*, *20*(2), 317–339, doi:10.1007/s10596-016-9558-3.

Formaggia, L. (1998), Data structures for unstructured mesh generation, in *Handbook of Grid Generation*, CRC Press. [Available at http://cat.inist.fr/?aModele=afficheN&cpsidt=16636291.]

Gong, B., M. Karimi-Fard, and L. J. Durlofsky (2006), An upscaling procedure for constructing generalized dual-porosity/dual-permeability models from discrete fracture characterizations, in *SPE Annual Technical Conference and Exhibition*, Soc. of Pet. Eng., doi:10.2118/102491-MS. [Available at https://www.onepetro.org/conference-paper/SPE-102491-MS.]

Gong, B., M. Karimi-Fard, and L. J. Durlofsky (2008a), Upscaling discrete fracture characterizations to dual-porosity, dual-permeability models for efficient simulation of flow with strong gravitational effects, *SPE J.*, *13*(01), 58–67, doi:10.2118/102491-PA.

Gong, B., G. Qin, C. Douglas, and S. Yuan (2011), Detailed modeling of the complex fracture network of shale gas reservoirs, *SPE Res. Eval. Eng.*, doi:10.2118/142705-MS. [Available at https://www.onepetro.org/conference-paper/SPE-142705-MS.]

Gong, Y., B. Li, and Z. Li (2008b), Immersed-interface finite-element methods for elliptic interface problems with nonhomogeneous jump conditions, *SIAM J. Numer. Anal.*, *46*(1), 472–495, doi:10.1137/060666482.

Hui, M.-H., B. Gong, M. Karimi-Fard, and L. J. Durlofsky (2007), Development and application of new computational procedures for modeling miscible gas injection in fractured reservoirs, in *SPE Annual Technical Conference and Exhibition*, Soc. of Pet. Eng., doi:10.2118/109686-MS. [Available at https://www.onepetro.org/conference-paper/SPE-109686-MS.]

Hui, M.-H., B. T. Mallison, and K.-T. Lim (2008), An innovative workflow to model fractures in a giant carbonate reservoir, in *International Petroleum Technology Conference*, doi:10.2523/12572-MS. [Available at https://www.onepetro.org/conference-paper/IPTC-12572-MS.]

Hui, M.-H., B. T. Mallison, M. H. Fyrozjaee, and W. Narr (2013), The upscaling of discrete fracture models for faster, coarse-scale simulations of IOR and EOR processes for fractured reservoirs, in *SPE Annual Technical Conference and Exhibition*, doi:10.2118/166075-MS. [Available at https://www.onepetro.org/conference-paper/SPE-166075-MS.]

Intersect (2014), High-resolution reservoir simulator. [Available at www.software.slb.com/products/foundation/Pages/intersect.aspx]

Karimi-Fard, M., L. J. Durlofsky, and K. Aziz (2004), An efficient discrete-fracture model applicable for general-purpose reservoir simulators, *SPE J.*, *9*(2), 227–236.

Karimi-Fard, M., B. Gong, and L. J. Durlofsky (2006), Generation of coarse-scale continuum flow models from detailed fracture characterizations, *Water Resour. Res.*, *42*, W10423, doi:10.1029/2006WR005015.

Kazemi, H., L. S. Merrill Jr., K. L. Porterfield, and P. R. Zeman (1976), Numerical simulation of water-oil flow in naturally fractured reservoirs, *SPE J.*, *16*(06), 317–326, doi:10.2118/5719-PA.

LeVeque, R. J. (2004), *Finite Volume Methods for Hyperbolic Problems*, Cambridge Texts Appl. Math., vol. 31, Cambridge Univ. Press. [Available at http://www.cambridge.org/fr/academic/subjects/mathematics/numerical-analysis/finite-volume-methods-hyperbolic-problems?format=PB.]

Li, L., and S. H. Lee (2008), Efficient field-scale simulation of black oil in a naturally fractured reservoir through discrete fracture networks and homogenized media, *SPE Reservoir Eval. Eng.*, *11*, 750–758, doi:10.2118/103901-PA.

Martin, V., J. Jaffré, and J. E. Roberts (2005), Modeling fractures and barriers as interfaces for flow in porous media, *SIAM J. Sci. Comput.*, *26*(5), 1667–1691, doi:10.1137/S1064827503429363.

Moinfar, A., A. Varavei, K. Sepehrnoori, and R. T. Johns (2014), Development of an efficient embedded discrete fracture model for 3d compositional reservoir simulation in fractured reservoirs, *SPE J.*, *19*(2), 289–303, doi:10.2118/154246-PA.

Oda, M. (1985), Permeability tensor for discontinuous rock masses, *Geotechnique*, *35*(4), 483–495, doi:10.1680/geot.1985.35.4.483.

Panfili, P., and A. Cominelli (2014), Simulation of miscible gas injection in a fractured carbonate reservoir using an embedded discrete fracture model, in *Abu Dhabi International Petroleum Exhibition and Conference*, Soc. of Pet. Eng., doi:10.2118/171830-MS.

Panfili, P., A. Cominelli, and A. Scotti (2013), Using Embedded Discrete Fracture Models (EDFMs) to Simulate Realistic Fluid Flow Problems, in *Second EAGE Workshop on Naturally Fractured Reservoirs*, Muscat, Oman. [Available at https://www.onepetro.org/conference-paper/SPE-171830-MS.]

Petrel (2014), E&P software platform. [Available at www.software.slb.com/products/platform/Pages/petrel.aspx.]

Pruess, K. (1992), Brief guide to the minc–method for modeling flow and transport in fractured media, *Tech. Rep. LBNL-32195*, Earth Sci. Div., Lawrence Berkeley Lab., Univ. of Calif., Berkeley.

Pruess, K., and T. N. Narasimhan (1985), A practical method for modeling fluid and heat flow in fractured porous media, *SPE J.*, *25*, 14–26, doi:10.2118/10509-PA.

Shimrat, M. (1962), Algorithm 112: Position of point relative to polygon, *Commun. ACM*, *5*(8), 434, doi:10.1145/368637.368653.

Si, H. (2015), Tetgen, a delaunay-based quality tetrahedral mesh generator, *ACM Trans. Math. Software*, *41*(2), 11:1–11:36, doi:10.1145/2629697.

Tatomir, A. B., A. Szymkiewicz, H. Class, and R. Helmig (2011), Modeling two phase flow in large scale fractured porous media with an extended multiple interacting continua method, *Comput. Model. Eng. Sci.*, *77*(2), 81–112, doi:10.3970/cmes.2011.077.081.

van Heel, A. P., P. M. Boerrigter, and J. J. van Dorp (2008), Thermal and hydraulic matrix-fracture interaction in dual-permeability simulation, *SPE Reservoir Eval. Eng.*, *11*(4), 735–749, doi:10.2118/102471-PA.

Warren, J. E., and P. J. Root (1963), The behavior of naturally fractured reservoirs, *SPE J.*, *3*(03), 245–255, doi:10.2118/426-PA.