SIAM J. Sci. Comput.
Vol. 31, No. 4, pp. 2688–2705

FLOW SIMULATION IN THREE-DIMENSIONAL DISCRETE FRACTURE NETWORKS[*]

JOCELYNE ERHEL[†], JEAN-RAYNALD DE DREUZY[‡], AND BAPTISTE POIRRIEZ[‡]

Abstract. In fractured rocks, fluid flows mostly within a complex arrangement of fractures. Both the fracture network structure and its hydraulic properties are determined at first order by the broad range of fracture lengths and densities. To handle the observed wide variety of fracture properties and the lack of direct fracture visualization, we develop a general and efficient stochastic numerical model for discrete fracture networks (DFNs) in a three-dimensional (3D) computational domain. We present an original conforming mesh generation method addressing the penalizing configurations stemming from close fractures and acute angles between fracture intersections. Flows are subsequently computed by using a mixed hybrid finite element (MHFE) method. The lack of direct fracture knowledge is treated by Monte-Carlo simulations requiring simulations with a large number of networks with various characteristics. We analyze the complexity in size and in time for the computation of flow in 3D DFNs meshed with our method and compare with the complexities for 2D rectangular domains meshed with a regular grid. We find out that complexity in size is similar whereas complexity in time is slightly larger for DFNs than for 2D regular domains.

Key words. discrete fracture network (DFN), mesh generation, mixed hybrid finite element method (MHFE)

AMS subject classifications. 65M50, 65M60, 65Y20, 65Z05

DOI. 10.1137/080729244

1. Introduction.

Numerical simulations in hydrogeology aim at finding the solution of the flow (and transport) equations in a complex, only partly identified, geologic system. Here we consider rocks where groundwater flow is channeled in fractures. Our model is based on discrete fracture networks (DFNs) and follows a stochastic approach.

1.1. Motivation.

The interest of fractured rocks for groundwater flow has grown in the last two decades in several respects, ranging from the storage of high-level nuclear wastes to water resources [8]. As fractures act as fast flow conduits, they determine the medium hydraulic properties like the flow pattern and the permeability. For rocks of very small permeability like crystalline rocks, the sole flow bearing structures are the fractures. Flow properties may be dominated by a few large fractures, by a dense network of small fractures, or by a combination of fractures of very different sizes [11], [12].

The nature of the flow pattern depends directly on the fracture length distribution and on the density of fractures. For dense fracture networks, flow will be well spread in the medium and could be advantageously modeled by continuous porous-like methods where fractures are implicitly taken into account by the permeability repartition of the equivalent porous medium. For sparse fracture networks, flow will be more likely channeled in the most permeable and accessible fractures and could be better modeled by discrete approaches where fractures are explicitly represented [23].

[*]Received by the editors July 3, 2008; accepted for publication (in revised form) March 10, 2009; published electronically June 25, 2009. This work was supported by the French government, with ACI-GRID project Hydrogrid and ANR-CIS project MICAS.
http://www.siam.org/journals/sisc/31-4/72924.html

[†]INRIA, Campus de Beaulieu, 35042 Rennes, France (Jocelyne.Erhel@inria.fr).
[‡]University of Rennes 1 (UMR CNRS 6118), Campus de Beaulieu, 35042 Rennes, France (Jean-Raynald.de-Dreuzy@univ-rennes1.fr, baptiste.poirriez@irisa.fr).

Flow properties depend also on the spatial variability of fracture aperture, which can be represented by a discrete approach. This method also provides ways to study flow and transport processes by numerical simulations [27] and may in turn give upscaling rules for permeability measurements, the flow pattern salient characteristics, and indications on hydraulic properties that cannot be directly observed on field tests.

Natural fractures occur on a broad range of scales. Because fractures cannot be observed in 3D, information on their characteristics comes from boreholes and outcrops. To address uncertainty in DFNs, stochastic approaches have been developed. The fracture shape is generally modeled by ellipses or polygons of varying aspect ratios [7], [22], [28]. Variable permeability can be modeled by a random distribution. We present in this paper a stochastic discrete approach working for a wide variety of fracture networks and adapted for both field and phenomenological studies.

1.2. Previous work. DFNs have a rather complex 3D geometry, since they are not 3D volumes but a set of 2D domains, with various orientations and intersecting each other. In flow computations, the rock matrix can be considered as impervious and flow is only simulated in the fractures. Simulation methods based on finite element methods face two problems. First, the number of cells in the mesh grows rapidly with the system size like in 3D models. Second, the 2D domains corresponding to the fractures display connection configurations difficult to mesh like small angles and points close to each other. There are two scales to consider in DFN: the 3D scale of the network and the 2D scale of each fracture. Two types of simulation methods were developed using either the two-scale structure in 2D fractures and 3D network fractures or a mesh generation of the full system at the network scale.

The first developed method consists of obtaining analytical relations between flows and heads within disk-shaped fractures through image theories, and of combining these analytical relations in a system of equations, giving the heads at the network scale [22]. Simpler approaches rely also on the two-scale fracture network structure. The mesh structure at the fracture scale is simplified by a network of monodimensional pipes between the fracture intersections and the fracture center, and the mesh structure at the network scale becomes a network of monodimensional pipes [7], [28]. This method solves the mesh generation and system size problems and respects the topological structure of the network. Its drawbacks are potential unrealistic flow patterns in the fracture, the difficult choice of the pipe transmissivities, and the impossibility to evaluate the uncertainties. Definition of pipe locations and transmissivities within the fractures has been more precisely quantified with use of flow solutions with the boundary element method within the fracture plan [13]. In this method, like in the method of [22], flow solving at the fracture scale provides relations between flows and heads that are used in a second step at the network scale. The boundary element method is, however, limited to homogeneous fractures and has only been applied to small fracture networks so far.

The second kind of simulation method consists of generating a mesh at the network scale and in using a finite element method for solving the flow equation. This strategy is implemented in several types of software like Rockflow and Fracman. It provides ways to introduce transmissivity heterogeneities within the fractures and to quantify uncertainties by reducing the characteristic mesh scale. Several solutions have been proposed for solving the problem of the connection configurations difficult to mesh or detrimental to numerical schemes. These configurations have been solved manually in each network through local modifications of the mesh [21], or removed from the network by small modifications of the fracture network structure [25].

1.3. Our contribution.

The latter approach can be integrated in a stochastic framework where several samples of DFNs are automatically generated and follow prescribed probability distributions. To our knowledge, most existing models do not allow broad fracture length distributions and broad fracture shape distributions occurring in natural fracture networks. In this article, we present a new flow simulation method based on an original mesh generator for general DFN structures.

Network generation is the first step of a global stochastic methodology. An original feature is the possibility to apply power-law fracture length distributions with a large range of possible exponents, correctly modeling both the natural broad range of scales and the variety of natural fractured sites and leading to a large number of fractures. It is also possible to define an heterogeneous permeability field, in order to take into account the spatial variability of fracture apertures.

In this paper, we focus on simple steady state flow. We assume that the rock matrix is impervious; classical equations are written in each fracture. In contrast to [26], we write also interface conditions for all intersections, in order to get a consistent system of equations. To solve these equations, a mixed finite element method can be used, where the difficulty is to express fluxes at the intersections. We use a hybrid method, different from [25], [29], but following [19], [20], where we replace fluxes by pressures at the edges. This approach is a very convenient and easy way to express mass continuity at the interfaces.

However, this method relies on mesh generation, providing conforming mesh structures. In general, a direct application of mesh generation does not succeed, because each fracture contains many intersections in all directions. In most cases, the quality of the mesh is poor, with small angles in the triangles [2], [26]. The objective of our mesh generation is to remove difficult connection configurations in the fractures. The idea is to apply a two-level discretization of the fracture intersections and boundaries. Intersections and boundaries are first referenced on a 3D regular grid of voxels (small cubes) common to all fractures. Then voxels to which elements of a fracture belong are projected back on the fracture plane. This discretization-projection of intersections and boundaries removes small angles and small segments so that fractures can be meshed with a high quality [2], [26]. However, for many generated networks, some boundary points are connected to more than two edges. In order to run stochastic simulations, a post-projection step is mandatory in order to remove these points.

When a mesh can be generated for many DFNs, it is possible to run many numerical experiments. In contrast to [26], we generate tests in a systematic way, using a wide range of parameters, several samples of Monte-Carlo simulations, and a wide range of mesh steps. These numerous tests are used to validate the method, to analyze convergence, and to study space and time complexity.

The paper is organized as follows. Section 2 describes the physical and numerical model. Then we develop our method to generate the mesh in section 3, describing our two-level process and our post-projection adjustments. Section 4 is devoted to software description and section 5 gives some experimental results.

2. Model.

Our model can be decomposed in three parts, related to three scientific fields: geology, hydraulics, and numerical analysis. We present these three components below.

2.1. Geological characteristics.

The model presented in this study is made up of ellipses identified by their length, shape, orientation, and position. The length is equal to their major axis and the shape noted e is defined by the ratio between their major axis and minor axis. The stochastic model addresses the issue of uncertainty in

Table 2.1

Random distributions for 3D discrete fracture networks. A random fracture is defined by four random variables: length, shape, position, and orientation.

Characteristic	Random distribution
length	power law
shape	uniform or constant
position	uniform
orientation	uniform

the data by generating fracture networks where the characteristics follow given probability distributions. The broad natural fracture length distribution can be correctly modeled by a power-law distribution such as

$$p(l)dl = \frac{1}{a-1}\frac{l^{-a}}{l_{\min}^{-a+1}}dl, \tag{2.1}$$

where $p(l)dl$ is the probability of observing a fracture with a length in the interval $[l, l+dl]$, l_{min} is the smallest fracture length, and a is a characteristic exponent [4], [10]. Values of the power-law length exponent a extrapolated from outcrop observations range between 2.5 and 5, traducing a wide variety of network structures. For a below 3, the network structure is dominated by the few largest fractures, whereas for a larger than 4, it is dominated by the smallest fractures. This original feature of our model leads to configurations with a large number of fractures spanning a large range of lengths. Without loss of generality, the fracture orientation and position distributions are taken as uniform, and ellipse shapes are either uniformly distributed or constant. The stochastic model is summarized in Table 2.1.

For simulation purposes, we introduce a cubic computational domain of characteristic size L in which the fracture networks are studied. The largest fracture length is of the order of L and the smallest fractures are of length close to l_{min}. The important parameter in the simulations is the scale range $[l_{min}, L]$ characterized by the scale ratio L/l_{min}.

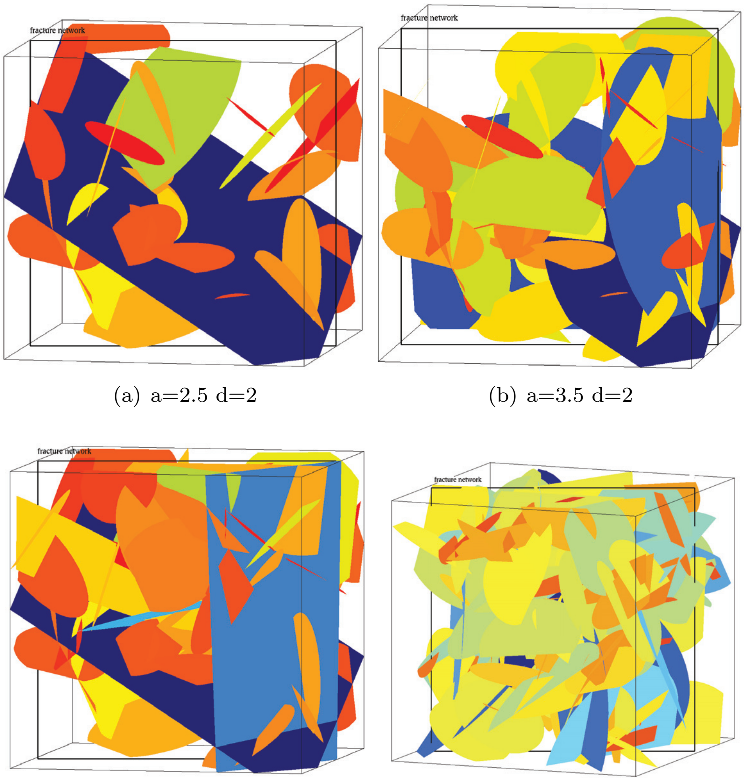
The global computational domain of the model is the network Ω, an open set composed of all the fractures and all the intersections between fractures. The boundary of Ω is composed of the borders of the cube and the edges of the ellipses. Let NF be the number of fractures. Each fracture is an open set $\Omega_f, f = 1, \dots NF$, an ellipse which is truncated if necessary by the faces of the cube. The boundary of a fracture is composed of the ellipse edge, the intersections with the cube faces, and the intersections with other fractures. Let NI be the number of intersections between fractures. Each intersection is a segment $S_k, k = 1, \dots NI$, and we denote by F_k the set of fractures with S_k on the boundary. Thus, we can write

$$\Omega = \bigcup_{f=1}^{NF} \Omega_f \cup \bigcup_{k=1}^{NI} S_k.$$

Fracture density is described by reference to the percolation threshold characterized by the critical number of fractures NF_c for which connection between the cube borders is reached. For power-law distributed fracture length, the critical density depends on the system size L [5], [6]. As this study focuses on connected fracture networks for flow simulations, we define the density as the ratio $d = NF/NF_c$. For example, $d = 2$ means that the system contains twice as many fractures than at threshold. Parameters of the model are summed up in Table 2.2.

Table 2.2
Parameters for 3D discrete fracture networks: exponent of power-law distribution, density of fractures, and scale ratio.

Parameter	Meaning	Range
a	power law	[2.5, 5]
d	density	$d \geq 1$
$L/l_{\min}$	scale ratio	$L/l_{\min} \geq 1$



(a) a=2.5 d=2

(b) a=3.5 d=2

(c) a=2.5 d=3

(d) a=4.5 d=3

Fig. 2.1. *Examples of of networks with various parameters: power-law exponent, density.*

Fracture characteristics have a critical influence on the fracture network topology. For example, the number of intersections NI increases with the fracture density d and decreases with the power-law exponent a. For ensuring a large applicability of our method, we cover in experiments a wide range of parameters, leading to 18 types of networks. Figure 2.1 displays four examples of networks used in this study.

2.2. Flow equations. Classical laws governing the flux in a porous medium are mass conservation and Darcy law. The main physical data is the permeability of the geological domain. In our model, the rock matrix surrounding the fractures is considered as impervious. Permeability of fractures can be highly variable. Indeed, they are closely related to apertures which vary spatially and can vary in time. In our current model, we consider a deterministic heterogeneous permeability field. In the future, we plan to consider also a random permeability field.

Since the rock matrix is impervious, we assume that mass conservation and Darcy laws are satisfied in each fracture, and we have to define conditions at the intersections. We first assume that there is no longitudinal flux in the intersections. If longitudinal flux is allowed, different interface conditions must be defined [24]. Then, we assume the continuity of the hydraulic head and the transversal flux at each intersection. These assumptions are quite classical and close the system of equations [29].

In order to write the laws in each fracture, we must define a projection from the 3D cube onto the fracture plane. Let (x, y, z) be the coordinates in the 3D cube and (x_f, y_f) the projected coordinates in the fracture Ω_f. The governing laws in each fracture are written

$$\begin{cases} v = -K\nabla h \text{ in } \Omega_f, \\ \nabla . v = s \text{ in } \Omega_f, \end{cases} \tag{2.2}$$

where the Darcy velocity v and the hydraulic head h are unknown, K is a given 2D permeability field, and s is a given source term. These equations are written in the fracture plane, using the 2D coordinates (x_f, y_f).

We assume that boundary conditions on the cube faces are either Dirichlet or Neumann. Let Γ_D be the part of the cube boundary with Dirichlet condition ($\Gamma_D \neq \emptyset$) and Γ_N be the part with Neumann condition. We also assume a Neumann zero flux condition on each ellipse edge Γ_f of fractures Ω_f.

Boundary conditions are written

$$\begin{cases} h = h_D \text{ on } \Gamma_D, \\ v.n = q_N \text{ on } \Gamma_N, \\ v.n = 0 \text{ on } \Gamma_f, \end{cases} \tag{2.3}$$

where h_D and q_N are prescribed values.

Continuity conditions in each intersection are written

$$\begin{cases} h_{k,f} = h_k, \text{ on } S_k, \forall f \in F_k, \\ \displaystyle\sum_{f \in F_k} v_{k,f} . n_{k,f} = 0 \text{ on } S_k, \end{cases} \tag{2.4}$$

where $h_{k,f}$ is the trace of the head and $n_{k,f}$ is the normal unit vector on the boundary S_k of the fracture Ω_f.

We assume that the set of equations (2.2)–(2.4) is well-posed in adequate functional spaces (it has a unique solution which depends continuously on the data).

2.3. Numerical method. Now, we have to define a spatial approximation in order to solve the physical model. We apply a mixed finite element (MFE) method for the following reasons. In general, finite element methods allow one to deal with complex geometries and to use locally refined meshes. In a mixed finite element method, the unknowns are hydraulic head and velocity; thus the velocity field is a good approximation, and useful for subsequent transport problems. Also this method guarantees both local and global mass conservation; therefore, MFE methods are very well-suited for solving flow problems. We do not define here the functional spaces and the variational weak mixed formulation, but refer to [25], [19], [20] for more details.

The first step is to define the mesh of the network. We require that the mesh is 2D in each fracture, in order to guarantee mass conservation in each fracture plane. We also rule that the mesh is conforming in each intersection. This means that the discretization of an intersection and of the boundaries is uniquely defined and that

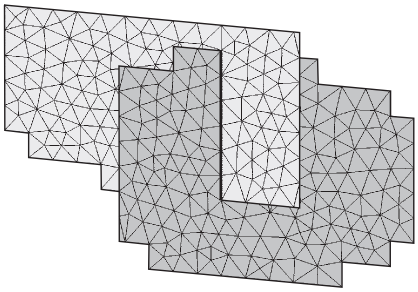


FIG. 2.2. *Two intersecting fractures with a triangular mesh: border edges, interior edges, and intersection edges.*

the discrete intersection is projected in the fracture plane of all fractures containing it. This constraint allows an easy definition of the discrete continuity conditions and allows us to apply classical results for MFE methods.

At the network level, boundaries and intersections are uniquely discretized with 3D coordinates. Then they are projected onto each fracture plane with 2D local coordinates, and each fracture can be meshed in 2D.

We assume that the 2D meshes are done with triangles. We use the hybrid version of the MFE method (MHFE method) where unknowns are mean head h_E in each triangle and mean head Φ_I on each edge [19]. For stationary flux equations, the MHFE method leads to a symmetric positive definite matrix, whereas the MFE method leads to a symmetric indefinite matrix. Also, in the context of our network of fractures, it is easy to handle intersection conditions with the MHFE method. Indeed, continuity of head-on intersection is trivial (only one unknown per edge); continuity of flux on intersection is easily accomplished by eliminating the velocity variable. In particular, it is possible and also sufficient to define locally in each element a discrete velocity field, whereas global unknowns are hydraulic heads h_E and Φ_I.

Let NT be the number of triangles and NE be the number of edges. For each triangle E, the three edges are locally numbered $i = 1, 2, 3$ and globally numbered I. For each edge I, the set of elements E containing I is denoted by T_I. The number of elements containing I varies: an edge on a boundary (cube or ellipse) belongs to only one triangle; an edge inside a fracture belongs to two triangles, and an edge on an intersection belongs generally to two fractures and four triangles. The different cases are illustrated by Figure 2.2, showing two fractures with their intersection.

We use the classical Raviart–Thomas basis functions $w_{E,j}$, defined in each 2D fracture and each triangle E, on each local edge j:

$$v_E = \sum_{j=1}^{3} v_{E,j} w_{E,j}.$$

Now, Dirichlet boundary conditions are given by $\Phi_I = h_D, I \in \Gamma_D$ and Neumann boundary conditions on the cube edge are given by $v_{E,i} = q_N, I \in \Gamma_N$, whereas Neumann boundary conditions on an ellipse edge are given by $v_{E,i} = 0, I \in \Gamma_f$, where E is the element containing I, with a local number i.

In order to eliminate the velocity and to derive a hybrid formulation, we write locally Darcy's law and express the velocity with hydraulic heads. Darcy's law $v = -K\nabla h$ is integrated $\nabla.v = s$ in each element E, using each basis function as a test function in the variational formulation; thus we get

$$\int_E K^{-1}v.w_{E,i} = -\int_E \nabla h.w_{E,i}.$$

The first term can be rewritten

$$\int_E K^{-1}v.w_{E,i} = \sum_j B_{i,j}^{(E)} v_{E,j} \text{ with } B_{i,j}^{(E)} = \int_E K^{-1}w_{E,j}.w_{E,i}.$$

Using the Green formula and the properties of the basis functions, the second term is rewritten:

$$-\int_E \nabla h.w_{E,i} = \int_E h\nabla.w_{E,i} - \int_{\partial E} hw_{E,i}.n_{\partial E} = h_E - \Phi_I.$$

Finally, we get the local Darcy's law in each element:

$$v_E = C^{(E)}(h_E u - \Phi_E), \tag{2.5}$$

where v_E is now the vector of the three components $v_{E,j}$, where $C^{(E)} = B^{(E)-1}$, Φ_E is the vector composed of the hydraulic heads at the three edges, and $u = (1,1,1)^T$. The matrices $B^{(E)}$ and $C^{(E)}$ are symmetric positive definite.

The discrete mass conservation equation is also obtained by integrating in each element:

$$\int_E \nabla.v = \int_E s, \text{ thus } \sum_i v_{E,i} = s_E.$$

Now, using the local Darcy's law (2.5), we get the global discrete mass conservation equation

$$Dh - R\Phi = s, \tag{2.6}$$

where h is the vector of the unknowns h_E and Φ is the vector of the unknowns Φ_I; the right-hand side s contains the source terms and the Dirichlet boundary conditions; the matrix D is diagonal and defined by $D_E = \sum_{i,j} C_{i,j}^{(E)}$; the matrix R is given by

$$\begin{aligned} &R_{E,I} = 0 \text{ if } I \notin E, \\ &R_{E,I} = \sum_j C_{i,j}^{(E)} \text{ if } I \in E. \end{aligned}$$

The discrete mass conservation is also written through each edge; the flux condition gives

$$\sum_{E\in T_I} v_{E,i} = 0, I \in \Omega_f; v_{E,i} = 0, I \in \Gamma_f; v_{E,i} = q_N, I \in \Gamma_N.$$

We deal now with intersection conditions (2.4). Since continuity of head is trivial here (the unknown Φ_I is unique on each conforming edge), only flux intersection conditions need to be written. Indeed, thanks to the conforming mesh, the mass conservation condition through an edge of a fracture is easily generalized through any edge of an intersection, and continuity of flux is written

$$\sum_{E\in T_I} v_{E,i} = 0, I \in S_k.$$

Using flux condition on any edge of a fracture or an intersection and using local Darcy's law (2.5), we get the global discrete Darcy's law:

$$-R^T h + M\Phi = q \tag{2.7}$$

with

$$\begin{cases} M_{I,J} = 0 \text{ if no element contains } I \text{ and } J, \\ M_{I,J} = C^{(E)}_{i,j},\ I \neq J, I \in E, J \in E, \\ M_{I,I} = \displaystyle\sum_{E \in T_I} C^{(E)}_{i,i}, \\ q_I = q_N \text{ if } I \in \Gamma_N, \\ q_I = 0 \text{ otherwise.} \end{cases}$$

By eliminating h, using the diagonal matrix D in (2.6), we get the linear system

$$A\Phi = b, \tag{2.8}$$

where the Schur complement matrix A and the right-hand side b are given by

$$\begin{cases} A = M - R^T D^{-1} R, \\ b = q + R^T D^{-1} s. \end{cases}$$

The head is then computed by $h = D^{-1}(R\Phi + s)$ and the velocity by (2.5).

The matrix A and right-hand side b are detailed below:

$$\begin{cases} A_{I,J} = M_{I,J} - R_{I,E} D_E^{-1} R_{J,E}, I \neq J, I \in \partial E, J \in \partial E \\ A_{I,I} = M_{I,I} - \displaystyle\sum_{E \in T_I} R_{I,E} {D_E}^{-1} R_{I,E}, \\ b_I = q_I + \sum_{E \in T_I} R_{I,E} {D_E}^{-1} s_E. \end{cases}$$

The linear system can be assembled from local matrices and vectors computed in each fracture. Let

$$\begin{cases} A^{(f)}_{I,J} = A_{I,J}, I, J \in \Omega_f, \\ A^{(f)}_{I,I} = \displaystyle\sum_{E \in T_I \cap \Omega_f} (C^{(E)}_{i,i} - R_{I,E} {D_E}^{-1} R_{I,E}), I \in \Omega_f, \\ b^{(f)}_I = q_I + \sum_{E \in T_I \cap \Omega_f} R_{I,E} {D_E}^{-1} s_E, \end{cases}$$

then $A = \sum_f A^{(f)}$ and $b = \sum_f b^{(f)}$. Therefore, it is possible to compute matrices locally in each fracture and to combine them in a global structure.

The matrix A is symmetric positive definite and is sparse. Sparse solvers, either direct or iterative, can be used to compute the solution. The size of the linear system is the number of edges in the network mesh and is roughly 1.5 times the number of triangles (about three edges for two triangles in 2D domains).

3. Mesh generation. In order to generate the mesh of the 3D DFN described above, we design and implement a new and original algorithm. To our knowledge, there is no existing software nor existing algorithm that would generate the mesh, which is neither 3D nor 2D nor a surface but a set of 2D meshes with a unique definition of their intersections. Because there is no specific direction of the flux, we aim at generating a mesh with a good aspect ratio, where all triangles are close to equilateral. Also, because we want to limit computational costs, we aim at reducing the number of elements in the mesh. Our algorithm is based on a two-level discretization. At the network level, each boundary and each intersection are discretized. Then the discrete boundaries and intersections of each fracture Ω_f are used to mesh the 2D domain. Therefore, we have to define a discrete projection from the 3D network onto each 2D fracture plane.

A natural approach would be to define each segment by a succession of discrete points and to project these points. However, this approach has several drawbacks. In some cases, it is not possible to generate a mesh; if the mesh can be generated, it has a tiny mesh size and many triangles, so that the linear system is huge; finally, it has very small angles, so that the aspect ratio is bad. For all these reasons, we define a new original approach to generate the mesh of DFNs. This approach is used in [2] and is described in [26].

The main idea is to define a 3D discretization of the boundaries and intersections. We introduce a 3D grid, with a given grid size Δx and with voxels as elementary units. The grid size is given relatively to $l_{\min}$, i.e., $\Delta x = \frac{absolute\ grid\ size}{l_{\min}}$. At the network level, boundaries and intersections are discretized by a set of voxels, instead of a set of points. Thus, at the network level, boundaries and intersections are now defined by 3D objects, which are projected onto fracture planes as before. The projected intersections and boundaries are now staircase lines with segments of length about Δx, so that small angles and small length size have disappeared. In each fracture, we can use any software to generate a 2D mesh, with Δx as initial mesh size. The resulting mesh has a good aspect ratio and a limited number of elements.

In contrast to [29], we keep a geometrical correspondence between the intersections. Let us consider an intersection between two fractures. The projections in each fracture are different, but the edges have the same 3D antecedent. Each edge in the intersection is thus uniquely defined, and can be identified in each fracture. Continuity conditions at the intersections can still be defined for each edge: same pressure for the corresponding edge in each fracture and continuity of flux through the edge. The mixed hybrid method remains somehow conforming, and we still get local and global mass conservation.

The projected fractures are no longer convex because of the discretization in voxels. An example of DFN is meshed using our new method, and the result can be seen in Figure 3.1. In this example, the 2D projected fracture (Figure 3.1(a)) is connected and the boundary is well identified, with intersections inside the boundary. Each boundary or intersection edge is connected to one or two other edges, and each boundary point is connected to one or two edges. The mesh can be generated and is of good quality (Figure 3.1(b)).

But in some cases, this property is no longer true, as can be seen in Figure 3.2. Indeed, it may happen that two points of the 3D boundary are discretized by the same voxel. In this case, the projected fracture may have several connected components or the boundary of the open set is no longer the set of boundary edges. Thus some points are connected to three edges and some edges are connected to three other edges. Some abnormal configurations are illustrated in Figure 3.2(a).

It is necessary to ensure some topological properties, each boundary or intersection edge must have only one or two neighbors and each boundary or intersection node must have only two adjacent edges. Otherwise, mesh generation can fail or elementary matrices can be singular, leading to a global singular matrix. Since these abnormal configurations may happen quite often in random 3D DFNs, it is mandatory to apply

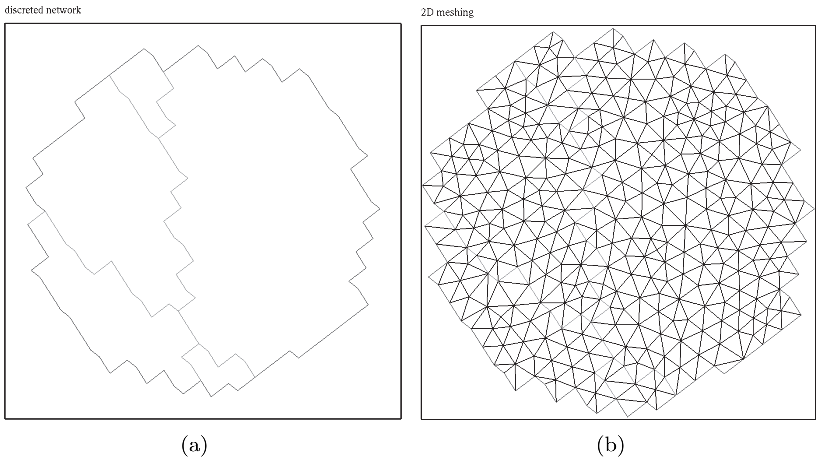


Fig. 3.1. *Projecting and meshing: example of a fracture; intersections are in light grey; the mesh has a good aspect ratio.*

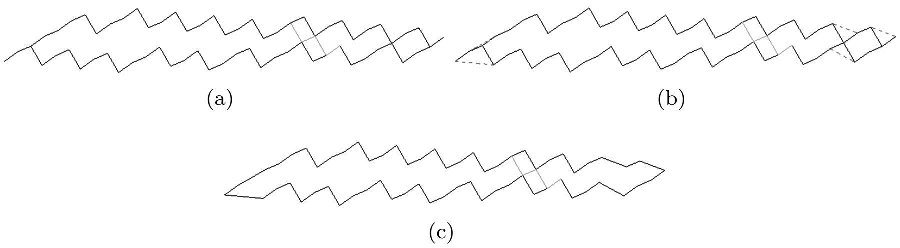


Fig. 3.2. *Local corrections before meshing: example of a fracture.* (a) *points connected to* 3 *or* 4 *edges;* (b) *local corrections;* (c) *after correction, each point is connected to* 2 *edges.*

a geometrical correction. In [26], this difficulty is not described, so networks used in the tests are specific and Monte-Carlo simulations are not feasible.

In order to respect these topological properties, we apply local corrections, which modify slightly the surface of the fracture. These local corrections aim at defining correct boundary and intersections. After the corrections, each edge is connected to one or two other edges and each node is connected to one or two edges. Examples of corrections are illustrated in Figure 3.2(b). After correction, the modified fracture has one connected component and the boundary is well defined, as illustrated in Figure 3.2(c). Thus we can guarantee that mesh generation will not fail, that aspect ratio is good, and that each elementary matrix is not singular, so that the global matrix is not singular.

4. Software. We have developed a complete software, written in C++, to generate random networks of fractures and to simulate flow in these networks. The software MP_FRAC is integrated in the scientific platform H2OLAB [15][1], which provides a user interface, a database for result compilation and analysis and visualization tools. The software MP_FRAC follows a modular and object-oriented approach, which provides a great flexibility by allowing an easy change of the algorithms and enhances

[1] http://h2olab.inria.fr/

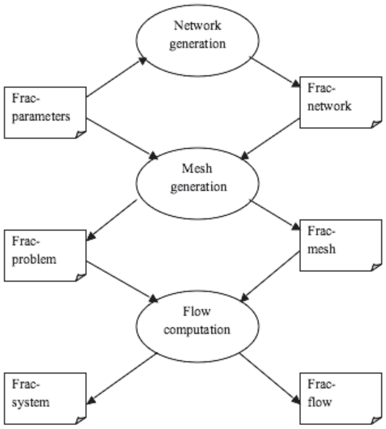


Fig. 4.1. *Computational modules and data structures for flow computations in discrete fracture networks.*

the maintenance. The code is organized in three modules, as depicted in Figure 4.1: network generation, mesh and problem generation, and flow computation.

The first module includes the random generation of the network, the definition of the geometry, the discretization of the intersections and boundaries, the global numbering of edges and points, and the projection of intersections and boundaries onto the fracture planes. This module builds the data structure "Network," which contains the geometrical data organized in a hierarchical way: geometrical information is stored at the various levels defining the network and fractures, the intersections and boundaries, the edges, and the points.

The second module makes use of the Network data structure to generate the mesh in each fracture, to number globally the edges, and to define the physical data. It builds the data structure "Mesh," which is also organized hierarchically with the same levels as Network with, in addition, the level defining the triangles. This module builds also the data structure "Pb-data," which contains the boundary conditions, the permeability field, the source term, the various physical data such as the gravitational constant, the density, etc.

Finally, the third module does the computation, by first computing the local matrices in each fracture and computing the global matrix and global right-hand side; then, it solves the sparse linear system, and computes the hydraulic head in each triangle and the velocity on each edge. This module uses previous data structures and builds the data structure "System," which contains the local matrices and right-hand sides, the global matrix and right-hand sides, the solution (hydraulic head on each edge), the hydraulic head in each triangle, and the velocity on each edge of each triangle. Then we check the result by a mass conservation verification.

The mesh generation is done by using a procedure extracted from the software FreeFem++ [18] and interfaced with a mixed finite element method. The local matrices and the velocity are computed by using adapted and rewritten procedures of the software Traces [19]. The sparse linear solver is a submodule which can be easily interfaced with free libraries. Currently, we have interfaced the direct solvers Umfpack [9] and Pspases [17], and the libraries Hypre [16] and Petsc [1].

5. Experimental results.

In order to illustrate the efficiency and robustness of our method, we generate a large number of test cases. We check the convergence order of the method by reducing the mesh size. Finally, we do a complexity analysis.

Table 5.1
Test generation: 180 *networks,* 1620 *systems.*

Parameter	Value
d	$2, 3$
$L/l_{\min}$	from 1 to 3
e	uniform
a	$2.5, 3.5, 4.5$
N_{ech}	10
Δx	from 0.01 to 0.09

5.1. Test generation. General networks have various parameters. We test our method on a set of parameters which covers a large set of DFNs, leading to a set of 18 network types. In all the tests, the permeability is chosen homogeneous and is equal to 1 everywhere. Since we use a Monte-Carlo method, for each set of parameters, we generate N_{ech} random samples, where $N_{ech} = 10$ is sufficient to validate the method. For each of the 180 networks, we vary the mesh step with 9 values, so that we get 1620 systems of meshed networks. Values are given in Table 5.1. We use the sparse direct solver Umfpack to compute the flow.

Since the solution for nonconnected networks is trivial ($v = 0$), we do not consider them. We test the connection after discretization. In fact, discretization may disconnect networks when intersections critical to connectivity (bottlenecks) are smaller than the discretization scale. Among the networks, we reject 254 nonconnected networks. Our method succeeds in generating the mesh and solving the linear system for all connected networks.

For most of the discretized networks, the post-projection corrections are essential. Indeed, if we remove the corrections, the method fails to generate a mesh or to solve the linear system. For the tests considered, the method without correction succeeds in only 222 cases. We have checked that the same problems occur when all fractures are disks, by taking a constant distribution of shapes. Again, the method without corrections succeeds in less than 300 cases, whereas our method succeeds in all cases. Therefore, it is not possible to run Monte-Carlo simulations without local corrections.

5.2. Validation, convergence, and order. For some particular cases, like a few orthogonal fractures, we check that the computed solution is as predicted by theory or analytical solution. For all cases, once we have solved flow equations, we check systematically that the computed fluxes satisfy global mass balance and local mass balance through each edge of the mesh. These verifications are a first step in validating the method.

Then, we analyze the convergence of the numerical solution when we reduce the mesh step. With a classical 2D or 3D domain, MHFE methods are of order 1; a priori error estimation can be derived in L^2 norm for the velocity [14]; when the domain is bounded, error estimation can also be written in L^1 norm. Here, we assume that the method applied to DFN with interface conditions is still of order 1 and that an error estimation can be written for the velocity at the intersections. Because we modify the geometry by projecting voxels and by local adjustments, there is no guarantee of convergence. Thus we check experimentally the behavior of numerical solutions. For each generated DFN, we realize a series of computations, each one with a different mesh step, ranging from 0.01 to 0.09. Since we do not know the exact solution, we use the simulation with the smallest mesh step as a reference to check the convergence. We use a physical criteria to check convergence, defined as the flow

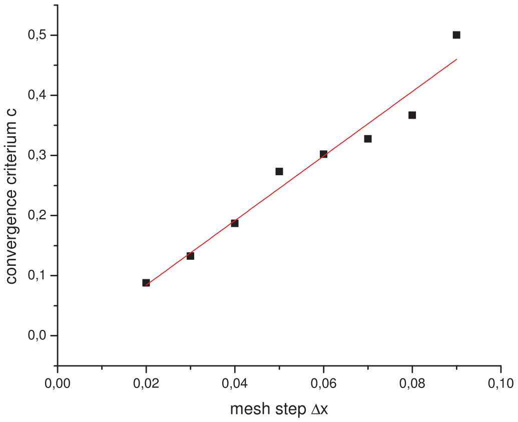


FIG. 5.1. *Convergence analysis: relative error on flux at intersections for a given mesh step; reference value is taken from the finest mesh. Mean values computed from all other meshed networks.*

crossing the intersections normalized by the cumulated intersections length. Let us define $L = \sum_{k=1}^{NI} l_k$, where l_k is the length of intersection S_k. Let us define, for a velocity v,

$$\Phi(v) = \sum_{k=1}^{NI} \int_{S_k} |v| dl.$$

We compute the scalar c defined by

$$c = \frac{|\Phi(v_{\Delta x}) - \Phi(v_{ref})|}{L},$$

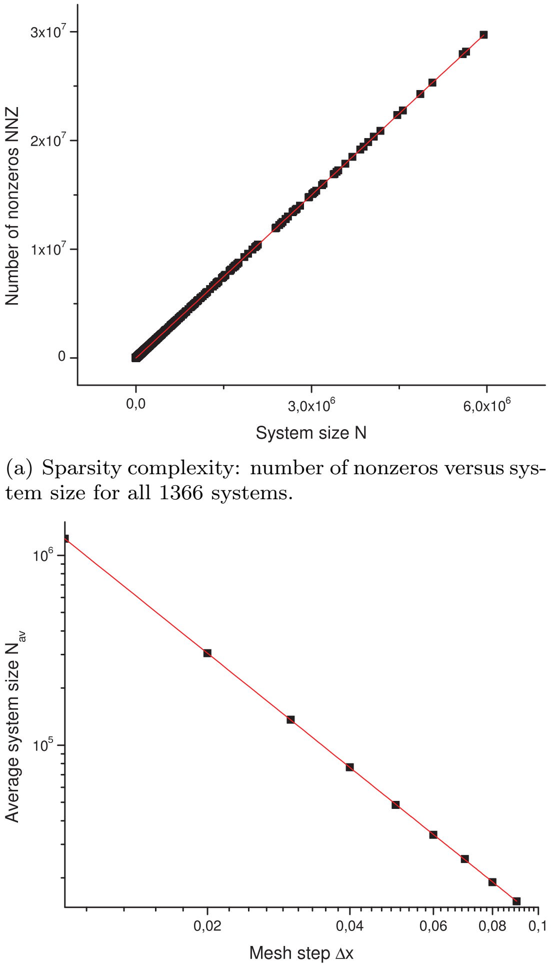
where v_{ref} is the velocity for the reference mesh. This criteria can be computed easily since

$$\Phi(v_{\Delta x}) = \sum_{I=1}^{N_{\Delta x}} |v_{E,i}| l_I,$$

where l_I is the length of edge I and $N_{\Delta x}$ is the total number of edges at intersections.

Using all the 1366 simulations, we calculate the mean of c for each mesh step. Figure 5.1 shows these means, where one point corresponds to about 150 networks with various parameters. We observe that c has a linear convergence, as can be expected if the method is of order 1.

5.3. Sparsity and size complexity. The objective here is to analyze how the size of the linear system scales with the various parameters. In a first step, we measure the sparsity of the matrix. The number of nonzero elements on a line is the number of neighbors of the associated edge plus one for the diagonal element. In the 2D case, with a triangular mesh, each interior edge has four neighbors and border intersections have only two neighbors, leading to a number of nonzeros NNZ roughly equal to $5N$, where N is the size of the matrix. In 3D DFNs, each fracture is 2D, but a special case occurs for intersections between fractures, since the edges on an intersection between two fractures have eight neighbors. Thus NNZ could be larger than $5N$ for some networks. In Figure 5.2(a), we plot NNZ versus N for all the systems. Clearly, we



(a) Sparsity complexity: number of nonzeros versus system size for all 1366 systems.

(b) Size complexity: average system size versus mesh step for all 1366 systems.

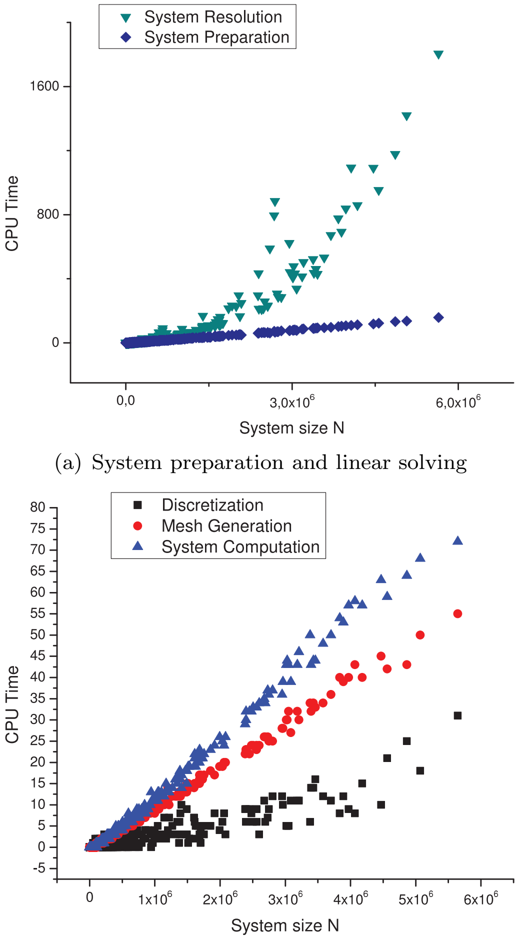
Fig. 5.2. *Linear system complexity in size.*

observe a factor $NNZ = 4.99N$, showing that the contribution of the intersections is smaller than the contribution of borders. Thus we get a sparsity complexity similar to the 2D case, although we deal with 3D DFNs.

In a second step, we analyze how the matrix size N scales with the mesh step Δx. In a classical 2D mesh, N behaves like Δx^{-2}, whereas N behaves like Δx^{-3} in 3D regular grids. For each mesh step Δx, ranging from 0.01 to 0.09, we compute the average size N_{av} of all generated networks. In Figure 5.2(b), we plot N_{av} versus Δx and find a power relation with an exponent -2, showing that N behaves like Δx^{-2}, still like the 2D case.

Therefore, the size and sparsity of the discrete system arising from flow computations in 3D discrete fracture networks have a behavior similar to a system arising from a 2D domain.

5.4. Time complexity. Now we analyze the complexity of the computation, by measuring the CPU time and taking the system size N as the main parameter of the



(a) System preparation and linear solving

(b) System preparation detail

Fig. 5.3. *Time complexity: CPU time versus system size for all generated networks.*

system. We measure the CPU time during two phases: the first phase concerns all the preparation before solving the linear system with a sparse direct solver, which is the second phase. We recall that we use a multifrontal solver, implemented in the library Umfpack. In Figure 5.3(a), we draw the CPU time versus the size for these two phases. Since we use a direct solver, the time for solving the system is not linear, but follows roughly a power law. We observe different tendencies, all of them with an exponent larger than 1.5, which is the exponent for flow computations in 2D regular grids [3]. Although the preparation phase includes more computations than a classical 2D MFE method, we can verify that the CPU time during this phase is linear in N and much smaller than the CPU time during the second phase. The details of the preparation phase are drawn on Figure 5.3(b), where we have split the time into three steps. The first step deals with 3D discretization, projection, and adjustments; the second step generates the mesh; the third step computes the matrix and right-hand-side. The first step, which is specific to our method, is the cheapest one, whereas computing the matrix is the most expensive.

6. Conclusion. In this paper, we consider 3D DFNs as a stochastic approach for modeling flow in natural fractured rocks. Flow simulations aim at characterizing flow patterns in such networks. This is achieved by applying an MHFE method on a flow model where continuity of mass and flux are imposed at the intersections between the fractures. The main difficulty is to generate a mesh, which must be 2D in each fracture and must be conforming at the intersections in order to apply the numerical method. We present a two-fold method, where the network is first discretized by voxels and the projected fractures are discretized by a classical mesh generator. We show that this method yields a mesh with a good aspect ratio, but requires adjustments in order to remove abnormal configurations after projection. Thanks to our method, it is possible to generate a mesh for a wide range of random DFN; we can vary the power-law exponent, the number of fractures, their orientation, and their shape. For all the networks tested, the mesh is of good quality, in the sense of a good aspect ratio, and a fairly large mesh size can be used. Therefore, it is possible to use uncertainty quantification methods such as Monte-Carlo simulations to study flow in fracture networks.

Although we deal with 3D structures, we show that the size complexity is similar to 2D computational domains. For any network, the size N of the linear system is the main parameter relevant to analyze CPU time; we find out that the CPU time of the preparation phase is linear with N and that most of the time is spent in solving the sparse linear system, with a complexity larger than $O(N^{1.5})$. We plan now to further analyze this behavior in order to reduce CPU time and to tackle even larger systems. We will run simulations with iterative solvers and will develop a parallel version. Another track of research is to relax the conforming constraint by applying a mortar methodology.

REFERENCES

[1] S. Balay, K. Buschelman, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, B. F. Smith, and H. Zhang, *PETSc Web page*, http://www.mcs.anl.gov/petsc (2001).

[2] A. Beaudoin, J.-R. de Dreuzy, J. Erhel, and H. Mustapha, *Parallel simulations of underground flow in porous and fractured media*, in Parallel Computing: Current and Future Issues of High-End Computing, G. Joubert, W. Nagel, F. Peters, O. Plata, P. Tirado, and E. Zapata, eds., Vol. 33 of NIC Series, NIC, Jülich, Germany, 2006, pp. 391–398.

[3] A. Beaudoin, J. Erhel, and J.-R. de Dreuzy, *A comparison between a direct and a multigrid sparse linear solvers for highly heterogeneous flux computations*, in Eccomas CFD 2006, vol. CD, 2006.

[4] E. Bonnet, O. Bour, N. Odling, P. Davy, I. Main, P. Cowie, and B. Berkowitz, *Scaling of fracture systems in geological media*, Rev. Geophys., 39 (2001), pp. 347–383.

[5] O. Bour and P. Davy, *Connectivity of random fault networks following a power law fault length distribution*, Water Resource Res., 33 (1997), pp. 1567–1583.

[6] O. Bour and P. Davy, *On the connectivity of three dimensional fault networks*, Water Resource Res., 34 (1998), pp. 2611–2622.

[7] M. C. Cacas, E. Ledoux, G. d. Marsily, A. Barbeau, P. Calmels, B. Gaillard, and R. Magritta, *Modeling fracture flow with a stochastic discrete fracture network: Calibration and validation.* 1. *The flow model*, Water Resource Res., 26 (1990), pp. 479–489.

[8] N. R. Council, *Rock Fractures and Fluid Flow*, National Academy Press, Washington, D.C., 1996.

[9] T. A. Davis, *Direct Methods for Sparse Linear Systems*, in SIAM Book Series on the Fundamentals of Algorithms, SIAM, Philadelphia, 2006.

[10] J.-R. de Dreuzy, P. Davy, and O. Bour, *Percolation parameter and percolation-threshold estimates for three-dimensional random ellipses with widely scattered distributions of eccentricity and size*, Phys. Rev. E, 62 (2000), pp. 5948–5952.

[11] J.-R. de Dreuzy, P. Davy, and O. Bour, *Hydraulic properties of two-dimensional random fracture networks following a power law length distribution:* 1-*effecive connectivity*, Water Resource Res., 37 (2001), pp. 2065–2078.

[12] J.-R. de Dreuzy, P. Davy, and O. Bour, *Hydraulic properties of two-dimensional random fracture networks following a power law length distribution: 2-permeability of networks based on log-normal distribution of apertures*, Water Resource Res., 37 (2001), pp. 2079–2095.

[13] W. S. Dershowitz and C. Fidelibus, *Derivation of equivalent pipe networks analogues for three-dimensional discrete fracture networks by the boundary element method*, Water Resource Res., 35 (1999), pp. 2685–2691.

[14] J. Douglas and J. Roberts, *Global estimates for mixed methods for second order elliptic methods*, Math. Comput., 44 (1985), pp. 39–52.

[15] J. Erhel, J.-R. de Dreuzy, A. Beaudoin, E. Bresciani, and D. Tromeur-Dervout, *A parallel scientific software for heterogeneous hydrogeology*, in Parallel Computational Fluid Dynamics 2007, I. H. Tuncer, U. Gulcat, D. R. Emerson, and K. Matsuno, eds., Lecture Notes in Computational Science and Engineering 67, Springer, 2009, pp. 39–48.

[16] R. Falgout, J. Jones, and U. Yang, *Numerical Solution of Partial Differential Equations on Parallel Computers*, The Design and Implementation of Hypre, a Library of Parallel High Performance Preconditioners, Springer-Verlag, Berlin, 2006, pp. 267–294.

[17] A. Gupta, *Recent advances in direct methods for solving unsymmetric sparse systems of linear equations*, ACM Trans. Math. Software, 28 (2002), pp. 301–324.

[18] F. Hecht, O. Pironneau, A. L. Hyaric, and K. Ohtsuka, *Freefem++*, 2nd ed., Technical report, Laboratory Jacques-Louis Lions, University Pierre et Marie Curie, 2007.

[19] H. Hoteit, J. Erhel, R. Mosé, B. Philippe, and P. Ackerer, *Numerical reliability for mixed methods applied to flow problems in porousmedia*, Comput. Geosci., 6 (2002), pp. 161–194.

[20] H. Hoteit, R. Mosé, B. Philippe, P. Ackerer, and J. Erhel, *The maximum principle violations of the mixed-hybrid finite-element method applied to diffusion equations*, Int. J. Numer. Methods Eng., 55 (2002), pp. 1373–1390.

[21] T. Kalbacher, R. Mettier, C. McDermott, W. Wang, G. Kosakowski, T. Taniguchi, and O. Kolditz, *Geometric modelling and object-oriented software concepts applied to a heterogeneous fractured network from the grimsel rock laboratory*, Comput. Geosci., 11 (2007), pp. 9–26.

[22] J. C. S. Long, P. Gilmour, and P. A. Witherspoon, *A model for steady fluid flow in random three-dimensional networks of disc-shaped fractures*, Water Resource Res., 21 (1985), pp. 1105–1115.

[23] J. C. S. Long, J. S. Remer, C. R. Wilson, and P. A. Witherspoon, *Porous media equivalents for networks of discontinuous fractures*, Water Resource Res., 18 (1982), pp. 645–658.

[24] V. Martin, J. Jaffré, and J. E. Roberts, *Modeling fractures and barriers as interfaces for flow in porous media*, SIAM J. Sci. Comput., 26 (2005), pp. 1667–1691.

[25] J. Maryska, O. Severýn, and M. Vohralík, *Numerical simulation of fracture flow with a mixed-hybrid fem stochastic discrete fracture network model*, Comput. Geosci., 8 (2004), pp. 217–234.

[26] H. Mustapha and K. Mustapha, *A new approach to simulating flow in discrete fracture networks with an optimized mesh*, SIAM J. Sci. Comput., 29 (2007), pp. 1439–1459.

[27] S. P. Neuman, *Trends, prospects and challenges in quantifying flow and transport through fractured rocks*, Hydrogeology J., 13 (2005), pp. 124–147.

[28] A. W. Nordqvist, Y. W. Tsang, C. F. Tsang, B. Dverstop, and J. Andersson, *A variable aperture fracture network model for flow and transport in fractured rocks*, Water Resource Res., 28 (1992), pp. 1703–1713.

[29] M. Vohralík, J. Maryska, and O. Severýn, *Mixed and nonconforming finite element methods on a system of polygons*, Appl. Numer. Math., 57 (2007), pp. 176–193.