A hybrid high-order method for flow simulations in discrete fracture networks

Florent Hédin, Géraldine Pichot and Alexandre Ern

Abstract We are interested in solving flow in large trimensional Discrete Fracture Networks (DFN) with the hybrid high-order (HHO) method. The objectives of this paper are: (1) to demonstrate the benefit of using a high-order method for computing macroscopic quantities, like the equivalent permeability of fracture rocks; (2) to present the computational efficiency of our C++ software, `NEF++`, which implements the solving of flow in fractures based on the HHO method.

1 The flow problem in fractured rocks

In fractured rocks, fluid flows mostly within a complex arrangement of fractures, classically modeled as a Discrete Fracture Network (DFN) [1, 2]. In the present reduced model, the fractures, denoted Ω_f, $f = 1, ..., N_f$, are distributed in a three-dimensional domain Ω and are modeled as ellipses whose position and orientation are evaluated from statistical laws given by geological studies [3, 4]. We consider single phase flow problems within these networks of fractures. As we are mainly interested in flow simulations in granite type rocks, a classical assumption is to consider the rock matrix as impervious. Figure 1 presents three examples of DFN in a cubic domain.

Florent Hédin
Inria & Université Paris-Est, CERMICS (ENPC), 2 rue Simone Iff, 75589 Paris, France, e-mail: florent.hedin@inria.fr

Géraldine Pichot
Inria & Université Paris-Est, CERMICS (ENPC), 2 rue Simone Iff, 75589 Paris, France, e-mail: geraldine.pichot@inria.fr

Alexandre Ern
Université Paris-Est, CERMICS (ENPC) & Inria, 77455 Marne-la-Vallée 2, France, e-mail: alexandre.ern@enpc.fr

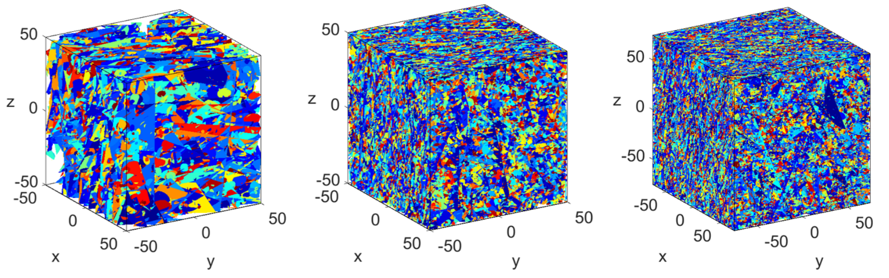

Fig. 1 (left) B1: 19,007 fractures; (center) B2: 152,399 fractures ; (right) B3: 508,338 fractures.

Let $\mathbf{x}$ be the local 2D coordinates of fracture Ω_f. Let N be the total number of intersections between fractures, I_k be the k$^{\text{th}}$ intersection, $k = 1, \ldots, N$, and F_k be the set of fractures containing I_k. In each fracture Ω_f, we assume that the governing equations for the hydraulic head scalar function p and for the flux per unit length function $\mathbf{u}$ are the mass conservation equation and Poiseuille's law [1]:

$$\nabla \cdot \mathbf{u}(\mathbf{x}) = f(\mathbf{x}) \quad \text{for } \mathbf{x} \in \Omega_f, \tag{1a}$$

$$\mathbf{u}(\mathbf{x}) = -\mathcal{T}(\mathbf{x})\, \nabla p(\mathbf{x}) \quad \text{for } \mathbf{x} \in \Omega_f. \tag{1b}$$

The parameter $\mathcal{T}(\mathbf{x})$ is a given transmissivity field (unit [m^2.s^{-1}]). The function $f \in L^2(\Omega_f)$ represents the sources/sinks. Additionally, continuity of the hydraulic head and continuity of the transversal flux apply at the intersections between the fractures [2, 5]:

$$p_{k,i} = p_k \quad \text{on } I_k, \forall f \in F_k, \tag{2a}$$

$$\sum_{i \in F_k} \mathbf{u}_{k,f} \cdot \mathbf{n}_{k,f} = 0 \quad \text{on } I_k, \tag{2b}$$

where $p_{k,i}$ is the trace of hydraulic head on I_k in fracture Ω_f, p_k is the unknown hydraulic head on the intersection I_k and $\mathbf{u}_{k,i} \cdot \mathbf{n}_{k,f}$ is the normal flux through I_k coming from fracture Ω_f, with $\mathbf{n}_{k,f}$ the outward normal unit vector of the intersection I_k with respect to the fracture Ω_f. Boundary conditions (BC) on the cube faces are of Dirichlet or Neumann type. For edges that belong to the border of the fractures but not to a cube face, a homogeneous Neumann BC is applied to express the imperviousness of the rock matrix.

2 The HHO method for solving flow in DFN

Several methods have been developed to solve flow in DFN in the recent years as detailed in the survey [6] and the references therein. The methods highly depend on the mesh strategy chosen to mesh the DFN [7, 8]. In our work, we keep the intersections explicitely. It implies a substantial work regarding the development of robust and efficient software in order to be able to mesh efficiently large networks with a good quality mesh [9]. In all our test cases, the mesh is generated with the software `BLSURF_FRAC` [10, 11] and the planar mesher is `BL2D`[12]. The data files generated by `BLSURF_FRAC` follow the description given by Appendix A in [13]. Here we consider the so-called conforming discretizations at the intersections between the ellipses but the software `BLSURF_FRAC` is also able to generate non-conforming discretizations as well. The advantage of keeping the intersections explicitely in the mesh generation is that it allows to attach unknowns to the edges and then continuity conditions (2) are easier to impose.

Among the methods that attach unknowns to the edges, let us cite the mixed-hybrid finite elements method (MHFEM), for conforming [14, 15, 16, 17, 5] or non-conforming [18, 19] discretizations at the intersections. More recently, a hybrid high-order (HHO) method has been developed [20, 21]. HHO is already used in many applications and has been recently used for fracture/matrix coupling [22]. HHO is closely related to Hybridizable Discontinuous Galerkin methods (HDG)[23]. The main advantages of HHO are (1) it allows general meshes (including polytopal cells and nonmatching interfaces), (2) it manages arbitrary polynomial face orders k, (3) it leads solve a linear system with only the unknowns at the edges and the matrix of this system is symmetric positive definite, (4) it delivers approximate solutions converging at order h^{k+1} in the energy norm and h^{k+2} in the L^2-norm (if full elliptic regularity holds) [20, 21]. Moreover, this HHO method is implemented in the open source library, `DiSk++` [24], which is a C++ template based library, both in the dimension and also in the finite element shapes. Notice that only the 2D feature of the `DiSk++` library is used in this study as the rock matrix is assumed impervious, however the dimensional templating offered by `DiSk++` will be very useful for future porous fractured rocks simulations.

3 Computation of the equivalent permeability with the HHO method

The goal of this section is to demonstrate the benefit of using a high-order method for computing upscaled quantities, like the equivalent permeability.

The equivalent permeability tensor is a macroscopic quantity of interest classically used by hydrogeologists for upscaling [25]. Its components can be derived from numerical simulations. Typically, the three diagonal components of the permeability tensor are given by applying permeameter boundary conditions in the directions x, y and z respectively. As we are rather interested in analyzing the performance of the HHO method to compute such macroscopic quantities, we focus here only on a flow in the direction x to derive the x-component of the permeability tensor defined as: $K_x = \dfrac{Q_{in,x}}{L\,\Delta h}$ for a cubic domain of size L, with $Q_{in,x}$ the input flux (units $\mathrm{m}^3.s^{-1}$) with respect to permeameter boundary conditions in the direction x.

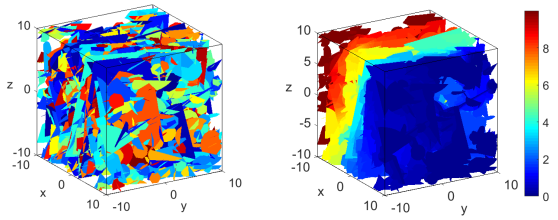

Fig. 2 (left) B0: 1,397 fractures, (right) Mean hydraulic head for permeameter BC

We propose to compute the equivalent permeability in the direction x of the small network $B0$ shown on Figure 2 (left). The domain is a cube of size $L = 20$. This network has $1,397$ fractures and $2,481$ intersections. We imposed a permeameter boundary condition in the direction x with a difference of hydraulic head of 10 m between the two opposite cube faces. The mean hydraulic head solution obtained with the MHFEM (Raviart-Thomas 0) for a mesh with $8,533,221$ edges is shown on Figure 2 (right). We compute K_x with the MHFEM RT0 and with the HHO method for face polynomial degrees $k = 0$, 1 and 2 and we compare the results. Table 1 presents the values of K_x as the mesh is refined.

DFN B0	Equivalent permeability K_x			
#edges	MHFEM RT0	HHO, $k = 0$	HHO, $k = 1$	HHO, $k = 2$
137,680	0.090929	0.090929	0.098410	0.099924
528,611	0.096663	0.096663	0.100556	0.101296
2,120,115	0.099615	0.099615	0.101507	0.101892
8,533,221	0.101032	0.101032	0.101943	0.102171
34,299,544	0.101696	0.101696	-	-

Table 1 *Test case B0: 1,397 fractures, computed values of the x-component of the equivalent permeability K_x with mesh refinement and different numerical methods.*

The simulations for $k = 1$ and $k = 2$ on the finer mesh are not available yet as they require a lot of computational resources. For the MHFEM RT0 method, the software that is used is a Matlab software, called `NEF-Flow` [11], developed at Inria and CNRS (France), and for the HHO method for DFN, the simulations are performed with the C++ software `NEF++`, developed at Inria and described in more details in the following section.

Figure 3 presents the equivalent permeability with respect to the number of degrees of freedom (dofs). For the MHFEM RT0 and the HHO method with $k = 0$, the number of dofs are equal to the total number of edges, denoted n_E. For the HHO method with $k = 1$, the number of dofs are $2\,n_E$. For the HHO method with $k = 2$, the number of dofs are $3\,n_E$. At low order ($k = 0$), the curves for K_x obtained with the HHO method and the MHFEM RT0 method almost superimpose, which was expected as the two methods are very close. The benefits of the increased orders of convergence are clearly seen from Figure 3 since the equivalent permeability is better approximated by a fixed number of DOFs if resorting to a higher-order method. This result shows that the exact solution has enough regularity to take advantage of the computational efficiency delivered by higher-order methods.

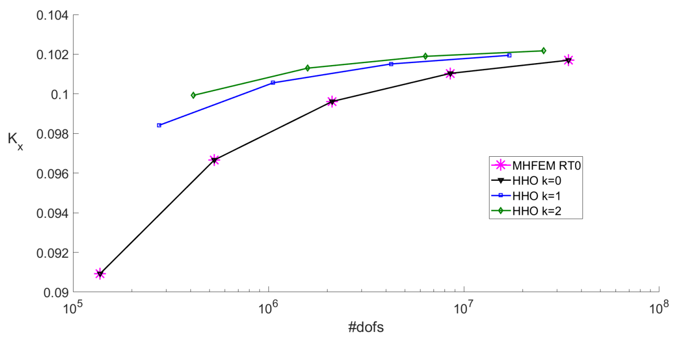

Fig. 3 Test case B0: x-component of the equivalent permeability tensor.

4 Performance obtained with the NEF++ software

Solving the flow problem (1) – (2) within large 3D DFNs requires robust and efficient software. The goal of this section is to present the C++ software we have developed at Inria, called NEF++, and based on the C++17 standard. NEF++ relies on the Eigen library, which is a C++ template library for linear algebra [26] and on the DiSk++ library for HHO. The linear systems can be solved with direct solvers or with iterative solvers like the preconditioned conjugate gradient or multigrid solvers. In NEF++, the following two direct solvers can be called: Pardiso from Intel MKL library [27] or SuiteSparse [28]. Both solvers support tasks parallelism, either using OpenMP or Intel TBB and support SIMD (Single Instruction, Multiple Data) vectorization.

We propose to solve flow in the three DFN B1, B2 and B3 shown on Figure 1. We imposed a permeameter boundary condition in the direction x with a difference of hydraulic head of 10 m between the two opposite cube faces. The transmissivity is taken as a constant per fracture and is different from one plane to another. We consider a cubic domain Ω of size $L = 100$ for $B1$ and $B2$ and $L = 150$ for $B3$. The network information about the geometry and the range of transmissivity values are given in Table 2.

	L	#fractures	#intersections	Range of transmissivity value [$m^2.s^{-1}$]
B1	100	19,007	28,727	[2.8e-06; 47.4]
B2	100	152,399	302,907	[3.4e-06; 20.33]
B3	150	508,338	1,031,231	[0.3e-05;25.8]

Table 2 *Details about the three DFN test cases $B1$, $B2$ and $B3$.*

For the larger linear systems (for $B1$ and $B2$ with $k = 1$ and $k = 2$ and for $B3$ with $k = 0, 1$ and 2), we are facing with the Intel Pardiso LLT solver some problems that we are currently investigating. On the contrary, we have no problem with the Intel Pardiso LU solver. In the following Tables 3, 4 and 5, the solver information (LLT or LU) will be given for each simulation. Moreover, depending on the requirements in computational resources, the simulations have been run either on a Intel Core i7 6 cores CPU laptop (denoted IC in the following Tables) or on a cluster node with 4 Intel Xeon E7-8890 processors and 1024 GiB of RAM (denoted IX in the following Tables).

Table 3 gives the computational time (for reading the mesh, assembling and solving the linear system) and peak memory of the `NEF++` software for $k = 0$ for the three test cases. For the B1 test case, we provide the results obtained with the LLT and LU solvers. With LU, the RAM memory requirements are higher than with LLT, as expected.

	#fractures	n_E	Read mesh	Assembling	Solving	Total time	RAM Peak	Solver	Run
B1	19,007	18,494,551	16.1 s	53.1 s	50.5 s	2 min	26.89 GiB	LLT	IC
B1	19,007	18,494,551	16.0 s	50.4 s	1 min 10 s	2 min17 s	30.59 GiB	LU	IC
B2	152,399	11,054,762	8.6 s	30.2 s	37.3 s	1 min 16 s	16.48 GiB	LLT	IC
B3	508,338	12,219,167	10.3 s	38.7 s	1 min 17s	2 min 7s	24.84 GiB	LU	IC

Table 3 *Performance obtained with `NEF++` for the HHO method with $k = 0$ on $B1$, $B2$ and $B3$.*

Table 4 and Table 5 give the computational time (for reading the mesh, assembling and solving the linear system) and peak memory of the `NEF++` software for $k = 1$ and $k = 2$ respectively for the three test cases. As shown by Tables 3, 4 and 5, despite $B3$ has less edges than $B1$, it takes more time to solve the associated linear system with a direct solver as it has more intersections (see Table 2).

	#fractures	n_E	Read mesh	Assembling	Solving	Total time	RAM Peak	Solver	Run
B1	19,007	18,494,551	20 s	5 min 57 s	3 min 36 s	9 min 53 s	105 GiB	LU	IX
B2	152,399	11,054,762	10 s	3 min 52 s	2 min 24 s	6 min 26 s	64 GiB	LU	IX
B3	508,338	12,219,167	12 s	4 min 42 s	3 min 52 s	8 min 47 s	78 GiB	LU	IX

Table 4 *Performance obtained with* `NEF++` *for the HHO method with* $k = 1$ *on* $B1$*,* $B2$ *and* $B3$.

	#fractures	n_E	Read mesh	Assembling	Solving	Total time	RAM Peak	Solver	Run
B1	19,007	18,494,551	20 s	13 min 48 s	11 min 4 s	25 min 12 s	204 GiB	LU	IX
B2	152,399	11,054,762	10 s	8 min 39 s	5 min 27 s	14 min 16 s	124 GiB	LU	IX
B3	508,338	12,219,167	12 s	9 min 48 s	12 min 9 s	22 min 9 s	153 GiB	LU	IX

Table 5 *Performance obtained with* `NEF++` *for the HHO method with* $k = 2$ *on* $B1$*,* $B2$ *and* $B3$.

5 Conclusion

The results in terms of computational time and accuracy we are currently obtaining with the `NEF++` software are very promising to handle in a near future the millions of fractures networks provided by external industrial partners. As the RAM requirements with direct solvers are quite large, we are currently investigating the use of iterative solvers. As emphasized in this paper, the HHO method has a strong potential, also for deriving upscaled quantities owing to its high-order feature. Moreover, as HHO naturally deals with general shape elements, non-conforming discretizations at the intersections between the fractures can be naturally handled in a conforming way. Finally, as a future work, we are interested in using the dimensional templating feature offered by the `DiSk++` library to solve flow in porous fractured rocks.

Acknowledgements The authors are grateful to the LabCom *fractory* (CNRS, Université de Rennes 1 and Itasca Consultants) for providing the geometry and transmissivity data for all the DFN considered in this paper. The authors warmly thank Nicolas Pignet (EDF) for many fruitful discussions related to `DiSk++`. Part of the simulations presented in this paper were carried out using the PlaFRIM experimental testbed, supported by Inria, CNRS (LABRI and IMB), Université de Bordeaux, Bordeaux INP and Conseil Régional d'Aquitaine (see https://www.plafrim.fr/).

References

1. V. Martin, J. Jaffré & J. E. Roberts, Modeling fractures and barriers as interfaces for flow in porous media, *SIAM Journal on Scientific Computing*, 26 (5), pp. 1667-1691, 2005.
2. J. Erhel, J. de Dreuzy, and B. Poirriez. Flow simulation in three-dimensional discrete fracture networks. *SIAM Journal on Scientific Computing*, 31(4):2688–2705, 2009.
3. P. Davy, R. Le Goc, C. Darcel, O. Bour, J.-R de Dreuzy & R. Munier, A likely universal model of fracture scaling and its consequence for crustal hydromechanics, *Journal of Geophysical Research: Solid Earth*, 115 (B10), 2010.
4. P. Davy, R. Le Goc & C. Darcel, A model of fracture nucleation, growth and arrest, and consequences for fracture density and scaling, *Journal of Geophysical Research: Solid Earth*, 118 (4), pp. 1393-1407, 2013.

5. J. Maryška, O. Severýn & M. Vohralík, Numerical simulation of fracture flow with a mixed-hybrid FEM stochastic discrete fracture network model, *Computational Geosciences*, 8, pp. 217-234, 2004.
6. A. Fumagalli, E. Keilegavlen, and S. Scialò. Conforming, non-conforming and non-matching discretization couplings in discrete fracture network simulations *J. Comput. Phys.*, 376, pp. 694–71, 2019.
7. J.D Hyman, S. Karra, N. Makedonska, C.W. Gable, S.L Painter, H.S. Viswanathan. dfnWorks: A discrete fracture network framework for modeling subsurface flow and transport. *Computers & Geosciences*, 84, pp. 10-19, 2015.
8. S. Berrone, S. Scialò and F. Vicini. Parallel Meshing, Discretization, and Computation of Flow in Massive Discrete Fracture Networks, *SIAM Journal on Scientific Computing*, 41:4, pp. C317-C338, 2019.
9. P. L. George, H. Borouchaki, F. Alauzet, P. Laug, A. Loseille & L. Maréchal, *Maillage, modélisation géométrique et simulation numérique 2 - Métriques, maillages et adaptation de maillages*, ISTE Editions, 412 pages (2018) (English translation to appear).
10. P. Laug & H. Borouchaki, BLSURF – Mesh Generator for Composite Parametric Surfaces – User's Manual, *Inria Technical Report*, RT-0235, 1999.
11. P. Laug and G. Pichot. Mesh Generation and Flow Simulation in Large Tridimensional Fracture Networks. https://hal.inria.fr/hal-02102811, 2019.
12. H. Borouchaki, P. Laug, and P. L. George. Parametric surface meshing using a combined advancing-front generalized-delaunay approach. *International Journal for Numerical Methods in Engineering*, 49(1-2):233–259, 2000.
13. J.-R de Dreuzy, G. Pichot, B. Poirriez & J. Erhel, Synthetic benchmark for modeling flow in 3D fractured media, *Computers & Geosciences*, 50, pp. 59-71, 2013.
14. P.A. Raviart & J. Thomas, A mixed finite element method for 2nd order elliptic problems, *Mathematical Aspects of the Finite Element Methods, Lectures Notes in Mathematics*, Springer, Berlin, 606, pp. 292-315, 1977.
15. D. N. Arnold & F. Brezzi, Mixed and nonconforming finite element methods: postprocessing, and error estimates, *ESAIM: M2AN*, 19, pp. 7-32, 1985.
16. F. Brezzi & M. Fortin, *Mixed and Hybrid Finite Element Methods*, Springer-Verlag, New York, 1991.
17. G. Chavent & J. E. Roberts, A unified physical presentation of mixed, mixed-hybrid finite elements and standard finite difference approximations for the determination of velocities in waterflow problems, *Advances in Water Resources*, 14 (6), pp. 329-348, 1991.
18. T. Arbogast, L. C. Cowsar, M. F. Wheeler & I. Yotov, Mixed finite element methods on nonmatching multiblock grids, *SIAM J. Numer. Anal.*, 37 (4), pp. 1295-1315, 2000.
19. G. Pichot, J. Erhel, and J.-R. de Dreuzy. A generalized mixed hybrid mortar method for solving flow in stochastic discrete fracture networks. *SIAM Journal on Scientific Computing*, 34(1):B86–B105, 2012.
20. D.A. Di Pietro, A. Ern, and S. Lemaire. An arbitrary-order and compact-stencil discretization of diffusion on general meshes based on local reconstruction operators. *Computational Methods in Applied Mathematics*, 14(4):461–472, 2014.
21. D.A Di Pietro and A. Ern. A hybrid high-order locking-free method for linear elasticity on general meshes. *Comp. Meth. Appl. Mech. Eng.*, 283, 1–21, 2015.
22. F. Chave, D.A. Di Pietro, and L. Formaggia. A hybrid high-order method for Darcy flows in fractured porous media. *SIAM J. Sci. Comput.*, 40, pp. A1063–A1094, 2018.
23. B. Cockburn, D.A. Di Pietro, and A. Ern. Bridging the hybrid high-order and hybridizable discontinuous galerkin methods. *ESAIM: M2AN*, 50(3):635–650, 2016.
24. M. Cicuttin, D.A. Di Pietro, and A. Ern. Implementation of Discontinuous Skeletal methods on arbitrary-dimensional, polytopal meshes using generic programming. *J. Comput. Appl. Math.*, 2017. http://hal.archives-ouvertes.fr/hal-01429292.
25. P. Renard & G. de Marsily, Calculating Equivalent Permeability, *Advances in Water Resources*, 20, pp. 253-278, 1997.
26. Gaël Guennebaud and Benoît Jacob et al., Eigen v3, http://eigen.tuxfamily.org, 2010.
27. https://software.intel.com/en-us/mkl, 2019.
28. http://faculty.cse.tamu.edu/davis/suitesparse.html, 2019.