New Polyhedral Discretisation Methods applied to the Richards Equation: CDO Schemes in Code Saturne

Jérôme Bonelle, Yvan Fournier, Charles Moulinec

New Polyhedral Discretisation Methods applied to the Richards Equation: CDO Schemes in *Code_Saturne*

Jérôme Bonelle
EDF R&D
6, quai Watier,
78401 Chatou, France

Yvan Fournier
EDF R&D
6, quai Watier,
78401 Chatou, France

Charles Moulinec
STFC
STFC Daresbury Laboratory,
Warrington, United Kingdom

Abstract

This paper shows an application of a vertex-based Compatible Discrete Operator (CDO) scheme to simulate the Richards equation on meshes using polyhedral cells. Second order spatial accuracy is reached for a verification test case using two meshes made of different types of polyhedral cells. A second validation test case dealing with a variably saturated soil inside a vertical column has been simulated, with three advected passive pollutants being released. Results are in good agreement with the reference for both types of mesh. MPI scalability has also been assessed at scale up to 98,304 cores of a Cray X30 for a 1.7 B cell mesh for the simulation of the aforementioned case, and nearly linear speed-up is observed. Finally the implementation of hybrid MPI-OpenMP has been tested on 2 types of Intel Xeon Phi Knights Landing co-processors, showing that the best configuration for the code is obtained when all the physical cores are filled by MPI tasks and the 4 hyperthreads by OpenMP threads.

1 Introduction

Advances in Computational Fluid Dynamics (CFD), multi-physics, numerical schemes, and High Performance Computing (HPC) make it possible to envisage extremely large simulations to model complex processes in complex configurations. However creating a mesh suitable for these complex geometries which will also produce accurate results might take more than 80% of the duration of a project, as many back and forth iterations between mesh generation and solver testing are required to derive the optimal input parameters for the solver. This is achieved by minimising mesh skewness and stretching in order to maintain accuracy. Reducing the mesh generation time is certainly important for academia, but crucial for industry, which usually expects quicker answers.

A way to reduce these back and forth iterations is to switch from the traditional Finite Volume (FV) [1] and Finite Element (FE) [2] approaches to methods which are more robust (in the sense they work fine on very distorted meshes) and more flexible (in the sense of the type of mesh allowed) while keeping a high accuracy.

Parallel mesh joining on-the-fly [4] is a very efficient tool to speed-up the mesh generation of complex geometries. This feature exists in *Code_Saturne* [5] as an in-house algorithm. A side effect of this flexibility is the addition of polyhedral cells at the interface between the two meshes to join if this interface is non-conforming. This yields a need to consider numerical schemes robust with respect to polyhedral meshes.

Research on discretisation schemes for polyhedral/distorted meshes (see Figure 1) has been worked on for many years [6, 7, 8], but has gained more and more attention in the last years [3]. Several methods have been recently revived or developed to be able to fulfil the following requirements, *i.e.* robustness, flexibility and accuracy. Besides the Compatible Discrete Operator (CDO) schemes [9, 10, 11, 12, 13],

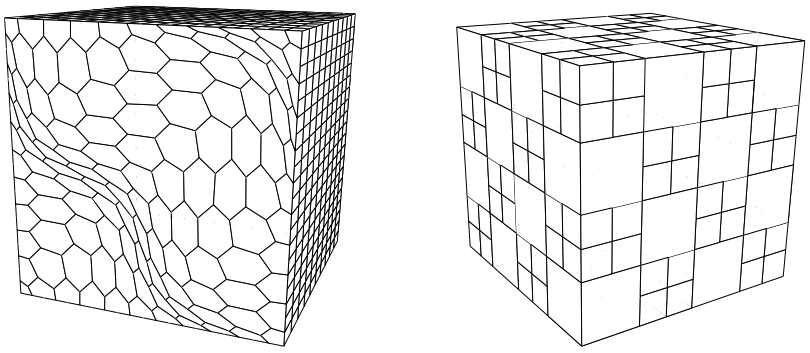

Figure 1: Examples of meshes used in the FVCA6 benchmark [3]. Left: Polyhedral cell mesh based on the extrusion of a 2-D polygon-based face. This type of mesh is denoted as PrG and the example shown corresponds to PrG10. Right: Polyhedral cell mesh with hanging nodes. This type of mesh is denoted as CB and the example shown corresponds to CB4.

several schemes sharing similar principles with the CDO schemes have been proposed, e.g. Mimetic Finite Differences [14, 15], Discrete Geometric Approach [16], Vertex Approximate Gradient scheme [17], SUSHI scheme [18]. All these schemes can be recast into the generic mathematical framework called Gradient schemes [19].

CDO schemes are very attractive because of their fully mathematical background, and their way of expressing exactly at discrete level topological laws such as conservation laws and differential operators, and of introducing approximations only when dealing with metric relations, such as closure or constitutive relations [10].

The implementation of CDO schemes is carried out in the *Code_Saturne* solver [5, 20], primarily developed by EDF, and open-source since 2007. The software uses the Finite Volume discretisation to simulate multi-physics CFD on meshes handling any type of cells. But as always for all the FV two-point discretisation methods, accuracy and robustness require good mesh quality. The FV version of the code relies on MPI to handle communications and OpenMP directives are added to speed-up the simulations. The code has been tested using over 3 million threads on MIRA (full machine) at Argonne (US DOE machine) for a 105 billion cell mesh and over 1.8 million threads on JUQUEEN (full machine) at Jülich (GE) for a 7 billion cell mesh, showing in both cases good strong scalability.

The ambition of this work and its novelty is to retain this level of parallelisation with the newly implemented CDO schemes, and to develop an hybrid MPI-OpenMP software showing good performance at scale, when keeping spacial second order accuracy for polyhedral cell meshes.

The paper is organised around the following sections. Section 2 shows a brief description of the CDO framework, Section 3 the extra parallelisation required to adapt a FV solver to CDO schemes. Section 4 introduces their application to the Richards equation, before describing in Section 5 the test case used for verification. Section 6 presents the validation of a vertex-based CDO scheme for a practical case and its performance on several architectures. Section 7 draws some conclusions and shows some directions for future work.

2 The CDO framework

CDO schemes belong to mimetic discretisations, also called structure-preserving or compatible discretisations. Their aim is to preserve the structural properties of the partial differential equations (PDEs) at discrete level by understanding the links between vector calculus, differential geometry and algebraic topology [21, 22, 23]. All these schemes rely on the seminal work of Kron [24] and Branin [25] for the simulation of large electrical networks. A detailed background about compatible spatial discretisations is given in [11, Chapter 2], where more references are to be found. The basic ingredients of the CDO

schemes and similar discretisations are: (1) a definition of the degrees of freedom (DoFs) according to their physical nature; (2) a definition of the discrete differential operators based on the fundamental theorem of calculus for the gradient, the Kelvin–Stokes theorem for the curl and the Gauss theorem for the divergence; (3) a clear distinction between topological laws, *i.e.* conservation laws, definitions of differential operators and, constitutive relations, such as the Fick law or the Fourier law, for instance, which rely on a metric.

2.1 Definition of the degrees of freedom

The leading idea is to consider all the mesh entities: vertices, edges, faces and cells. A vertex is denoted by v and the set of all vertices by V, an edge by e (E), a face by f (F) and a cell by c (C). Generically, we denote by $A_x := \{a \in A | a \subset \partial x\}$ if x has a dimension larger than a and, otherwise, $A_x := \{a \in A | x \subset \partial a\}$. ∂ denotes the boundary of a geometrical entity. The DoFs are organised as follows:

- DoFs physically related to a potential field are attached to mesh vertices and evaluated at each vertex location,
- DoFs physically related to a circulation field are attached to mesh edges and evaluated along each edge length,
- DoFs physically related to a flux field are attached to mesh faces and evaluated across each face area,
- DoFs physically related to a density field are attached to mesh cells and evaluated inside each cell volume.

Denoting by p (resp. $\underline{u}$, $\underline{\phi}$, s) a potential (resp. circulation, flux, density) field, the definition of each DoF reads:

$$\forall v \in \mathrm{V}, \qquad \mathbf{p}_v := p(v), \tag{1}$$

$$\forall e \in \mathrm{E}, \qquad \mathbf{u}_e := \int_e \underline{u} \cdot \underline{t}_e, \tag{2}$$

$$\forall f \in \mathrm{F}, \qquad \boldsymbol{\phi}_f := \int_f \underline{\phi} \cdot \underline{n}_f, \tag{3}$$

$$\forall c \in \mathrm{C}, \qquad \mathbf{s}_c := \int_c s, \tag{4}$$

where $\underline{t}_e$ is the unit tangent vector to e fixing its orientation and $\underline{n}_f$ is the unit normal vector to f fixing also its orientation; see Figure 2. $\mathbf{p}$ (resp. $\mathbf{u}$, $\boldsymbol{\phi}$ and $\mathbf{s}$) is an array of real values of size the number of mesh vertices (resp. edges, faces and cells) which belongs to the space of DoFs denoted by $\mathcal{V}$ (resp. $\mathcal{E}$, $\mathcal{F}$, $\mathcal{C}$). $\mathbf{p}_v$ (resp. $\mathbf{u}_e$, $\boldsymbol{\phi}_f$ and $\mathbf{s}_c$) corresponds to the entry in the DoF array related to the vertex v (resp. edge e, face f and cell c). Maps are used to define the DoFs. They are called *reduction maps* (or de Rham's maps) denoted by $\mathsf{R}_{\mathcal{X}}$ where $\mathcal{X}$ is any of the DoF spaces $\mathcal{V}$, $\mathcal{E}$, $\mathcal{F}$ or $\mathcal{C}$.

To avoid technicalities, we assume that fields are smooth enough functions so that the definition of the DoFs is well-defined and single-valued; see [11, § 3.1.3] for a detailed definition of the requirements on functional spaces.

2.2 Discrete differential operators

The starting relation to define the discrete gradient operator GRAD is $\int_e \underline{\mathrm{grad}}(p) \cdot \underline{t}_e = p(v_2) - p(v_1)$ for every edge $e \in \mathrm{E}$ with a starting vertex v_1 and an ending vertex v_2, leading to

$$\mathsf{GRAD} : \mathcal{V} \to \mathcal{E}, \quad \mathsf{GRAD}(\mathbf{p})|_e := \sum_{v \in \mathrm{V}_e} \pm \mathbf{p}_v, \ \forall e \in \mathrm{E}. \tag{5}$$

$\mathbf{p}_v$ is multiplied by $+1$ if $\underline{t}_e$ points towards the vertex v otherwise $\mathbf{p}_v$ is multiplied by -1.

The starting relation to define the discrete curl operator CURL is $\int_f \underline{\mathrm{curl}}\,\underline{u}\cdot\underline{n}_f = \int_{\partial f}\underline{u}\cdot\underline{t}_{\partial f}$ for every face $f\in\mathrm{F}$, leading to

$$\mathsf{CURL}:\mathcal{E}\to\mathcal{F},\quad \mathsf{CURL}(\mathbf{u})|_f := \sum_{e\in\mathrm{E}_f}\pm\mathbf{u}_e,\ \forall f\in\mathrm{F}. \tag{6}$$

$\mathbf{u}_e$ is multiplied by $+1$ if $\underline{t}_e\cdot\underline{t}_{\partial f}>0$ otherwise $\mathbf{u}_e$ is multiplied by -1.

The starting relation to define the discrete divergence operator DIV is $\int_c \mathrm{div}(\underline{\phi}) = \int_{\partial c}\underline{\phi}\cdot\underline{n}_{\partial c}$ for all cell $c\in\mathrm{C}$, leading to

$$\mathsf{DIV}:\mathcal{F}\to\mathcal{C},\quad \mathsf{DIV}(\boldsymbol{\phi})|_c := \sum_{f\in\mathrm{F}_c}\pm\boldsymbol{\phi}_f,\ \forall c\in\mathrm{C}. \tag{7}$$

$\boldsymbol{\phi}_f$ is multiplied by $+1$ if $\underline{n}_f$ points outward otherwise $\boldsymbol{\phi}_f$ is multiplied by -1.

From the 3 definitions (5), (6), (7), the following properties (cf. [11, p. 24] for the proofs) can be derived for the compatible discretisations. They correspond to the discrete counterpart of the fundamental properties of the differential operators.

Proposition 2.1 (Complex)**.**

$$\mathsf{CURL}\cdot\mathsf{GRAD}=0_{\mathcal{F}},\quad \mathsf{DIV}\cdot\mathsf{CURL}=0_{\mathcal{C}}.$$

Proposition 2.2 (Exactness)**.** *Under the assumption that the computational domain Ω is simply connected and its boundary $\partial\Omega$ is connected, the following identities between the range and the kernel of the discrete differential operators hold:*

$$\mathrm{Im}(\mathsf{GRAD})=\mathrm{Ker}(\mathsf{CURL}),\quad \mathrm{Im}(\mathsf{CURL})=\mathrm{Ker}(\mathsf{DIV}).$$

Proposition 2.3 (Commutation)**.**

$$\mathsf{R}_{\mathcal{E}}(\underline{\mathrm{grad}}) = \mathsf{GRAD}(\mathsf{R}_{\mathcal{V}}), \tag{8}$$

$$\mathsf{R}_{\mathcal{F}}(\underline{\mathrm{curl}}) = \mathsf{CURL}(\mathsf{R}_{\mathcal{E}}), \tag{9}$$

$$\mathsf{R}_{\mathcal{C}}(\mathrm{div}) = \mathsf{DIV}(\mathsf{R}_{\mathcal{F}}). \tag{10}$$

Proposition 2.3 states that there is no consistency error introduced during the discretisation of the differential operators; see [21, 26]. Eq. (10) is also used in the context of FV schemes and is a key point to correctly implement conservation laws.

2.3 Dual mesh

CDO schemes, as opposed to some other compatible discretisations use the initial mesh, called the *primal mesh*, and also a second mesh, called the *dual mesh*. Primal and dual meshes do not play the same role. The problem data (boundary conditions, definitions of physical properties, initialisation) are set on the primal mesh. The dual mesh plays a role in the construction of the numerical scheme and is not seen by the end-user. There are several ways to build this mesh following these two leading principles: (1) one-to-one pairing between primal and dual entities. For instance, a primal mesh vertex $v\in\mathrm{V}$ is attached to a dual mesh cell $\tilde{c}(v)$, a primal mesh edge $e\in\mathrm{E}$ to a dual mesh face $\tilde{f}(e)$ and so on; (2) the arbitrary orientation $\underline{t}_e$ related to a primal mesh edge $e\in\mathrm{E}$ is transferred to the normal of the associated dual face $\underline{n}_{\tilde{f}(e)}$ (*i.e.* $\underline{t}_e\cdot\underline{n}_{\tilde{f}(e)}>0$) and the arbitrary orientation $\underline{n}_f$ related to a primal mesh face $f\in\mathrm{F}$ is transferred to the tangential vector related to the associated dual edge $\underline{t}_{\tilde{e}(f)}$; see Figure 2. By default, CDO schemes use *barycentric* (and not *voronoï*) dual meshes in order to handle quite general polyhedral meshes. The discrete differential operators $\widetilde{\mathsf{GRAD}}$, $\widetilde{\mathsf{CURL}}$ and $\widetilde{\mathsf{DIV}}$ are defined following the same design principles as for the primal mesh. The same properties hold. The only additional care in comparison with the primal case is for the boundary conditions; see [11] for more details.

The dual mesh enables to discretise closure laws without additional interpolation. For instance, considering the Fourier law as:

$$\underline{\phi} = -\underline{\underline{\kappa}}\,\underline{\mathrm{grad}}(p) \tag{11}$$

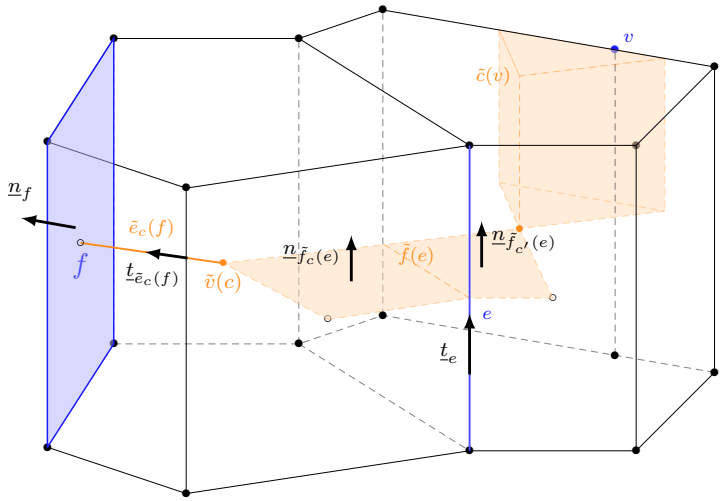

Figure 2: Primal (in blue) and dual (in orange) meshes and their related entities.

where $\underline{\underline{\kappa}}$ is the conductivity tensor, if the gradient is discretised along primal mesh edges then a *natural* way of discretising the induced flux is across dual faces which are in one-to-one pairing with these primal edges.

2.4 Discrete Hodge operators

The discrete Hodge operator serves as a link between quantities defined on primal entities and quantities defined on its dual entities. Contrary to discrete differential operators, there is no unique definition of discrete Hodge operators. A set of minimal requirements has been devised to ensure the convergence of the scheme. In the case of elliptic problems, two main properties are stated (1) stability and (2) $\mathbb{P}_0$-consistency (*i.e.* no discretisation error for constant vector field). Thus, several discrete Hodge operators fulfilling these requirements can be built leading to different CDO schemes but belonging to the same mathematical framework. Another straightforward consequence is that discrete Hodge operators introduce a consistency error contrary to discrete differential operators. For instance, $\mathsf{H}^{\nu\tilde{c}}$ interpolates density DoFs on the dual mesh starting from potential DoFs on the primal mesh. Similarly, $\mathsf{H}^{\varepsilon\tilde{f}}$ is the discrete Hodge operator between DoFs attached to primal edges and those attached to dual faces, *i.e.* a interpolation of dual fluxes starting from primal circulations. Thus, the discrete counterpart of (11) writes

$$\boldsymbol{\phi} = -\mathsf{H}_{\kappa}^{\varepsilon\tilde{f}} \cdot \mathsf{GRAD}(\mathbf{p}) \quad \forall \mathbf{p} \in \mathcal{V}, \tag{12}$$

where the subscript κ indicates that the discrete Hodge operator is weighted by the conductivity tensor. A generic way to define the discrete Hodge operator is to consider a reconstruction operator such that:

$$\left[\mathbf{a}, \mathsf{H}_{\kappa}^{\varepsilon\tilde{f}}(\mathbf{b})\right] := \int_{\Omega} \underline{\mathrm{L}}_{\mathcal{E}}(\mathbf{a}) \cdot \underline{\underline{\kappa}} \cdot \underline{\mathrm{L}}_{\mathcal{E}}(\mathbf{b}), \qquad \forall \mathbf{a}, \mathbf{b} \in \mathcal{E}, \tag{13}$$

where for all $\underline{x} \in \Omega$, $\underline{\mathrm{L}}_{\mathcal{E}}(\mathbf{a})(\underline{x}) = \sum_{e \in \mathrm{E}} \mathbf{a}_{\mathrm{E}} \underline{\ell}_e(\underline{x})$. The set $\{\underline{\ell}_e\}_{e \in \mathrm{E}}$ corresponds to reconstruction functions, or shape functions in the FE context, associated to an entity; an edge in this example, cf. [27] for more details. If two DoF spaces are considered, *i.e.* $\mathcal{X}$ and its DoF space in duality $\widetilde{\mathcal{Y}}$, then, for all $\mathbf{a} \in \mathcal{X}$ and $\mathbf{b} \in \widetilde{\mathcal{Y}}$, $[\mathbf{a}, \mathbf{b}]_{\mathcal{X}\widetilde{\mathcal{Y}}} = \sum_{x \in X} \mathbf{a}_x \mathbf{b}_x$ denotes the duality product.

The focus of this paper is on discrete Hodge operators based on the reconstruction operators introduced in the Discrete Geometric Approach (DGA) [28]. Other choices are possible such as the reconstructions devised in [18] and [29]; see also [11] for more details. The focus is narrowed down to two types of discrete Hodge operators, *i.e.* $\mathsf{H}^{\varepsilon\tilde{f}}$ and $\mathsf{H}^{\nu\tilde{c}}$.

2.5 Example of a CDO scheme for a pure diffusion problem

Considering homegeneous Dirichlet boundary conditions, a potential $p \in H^1(\Omega)$ and a source term $s \in L^2(\Omega)$, the model equation is

$$-\operatorname{div}(\underline{\underline{\kappa}} \cdot \underline{\operatorname{grad} p}) = s, \quad \text{on } \Omega. \tag{14}$$

Introducing intermediate unknowns, Eq. (14) can be recast into three elementary equations:

$$\operatorname{div}(\underline{\phi}) = s, \quad \underline{\phi} = -\underline{\underline{\kappa}}\,\underline{g}, \quad \underline{g} = \underline{\operatorname{grad} p}. \tag{15}$$

The left relation states the conservation of the quantity, the middle one corresponds to the closure relation where physical modelling is introduced and the right equation simply gives the definition of the gradient. The left and right equations are called *topological* relations since they are independent of the problem at stake, *i.e.* the geometry and the model for instance, whereas the middle one depends on the modelling and the type of problem.

Considering a CDO vertex-based discretisation, the discrete counterpart of Eq. (15) writes:

$$\widetilde{\mathsf{DIV}}(\boldsymbol{\phi}) = \mathsf{R}_{\widetilde{\mathcal{C}}}(s), \quad \boldsymbol{\phi} = -\mathsf{H}_{\kappa}^{\tilde{\varepsilon}\tilde{\mathcal{F}}}(\mathbf{g}), \quad \mathbf{g} = \mathsf{GRAD}(\mathbf{p}). \tag{16}$$

Owing to the properties stated in Sections 2.2 and 2.4, approximations are only introduced in the middle equation in (16), *i.e.* where the physical modelling also introduces some approximations. Combining the 3 equations listed in (16) yields:

$$-\widetilde{\mathsf{DIV}} \cdot \mathsf{H}_{\kappa}^{\tilde{\varepsilon}\tilde{\mathcal{F}}} \cdot \mathsf{GRAD}(\mathbf{p}) = \mathsf{R}_{\widetilde{\mathcal{C}}}(s) \tag{17}$$

3 Parallelisation

3.1 Base parallelisation approach

Although the CDO type schemes represent new developments in *Code_Saturne*, we seek to leverage the code parallelisation used for the base 2-point flux FV schemes, so as both to allow future simultaneous use of different schemes for incremental coupling of specific physical models, and mostly to benefit from the significant efforts already invested in this area.

The base Finite Volume scheme is cell-centered, and uses classical MPI-based parallelism with cell-based partitioning and ghost cells. Interior faces and vertices on rank boundaries are duplicated, and shared-memory (OpenMP) parallelism may be used inside each domain.

Global (partition-independent) numbering and a "block" distribution based on global number ranges is used for parallel checkpoint/restart and post-processing I/O as well as global mesh modifications. An optimised partitioning based on a graph bisection (PT-SCOTCH or ParMETIS) or space-filling-curve (Morton or Hilbert) approach is used for most compute operations. Data can be swapped from one distribution scheme to the other, so that except for initial mesh import to a native low-level format, all operations are distributed. This first level of parallelism is naturally based on MPI, and includes some aspects specific to *Code_Saturne*:

- Base halo contains cells with face-adjacency for main scheme (see Figure 3), optional extended halo with vertex-adjacency for least-squares gradient reconstruction;
- Halos also used for true implicit periodicity handling, subdivided by transformation to allow rotating vectors when synchronising;
- A second (reduction-based) global numbering is also used for optional external linear algebra libraries such as PETSc or HYPRE.

A loop-based OpenMP approach is also used for a second layer of parallelism. On a small number of MPI ranks, this does not usually provide high performance (as the code is mostly memory-bound, with low computational intensity in most operations, and some less compute-intensive parts are not threaded) but with OpenMP's usually lower overheads, Hybrid MPI-OpenMP performance becomes

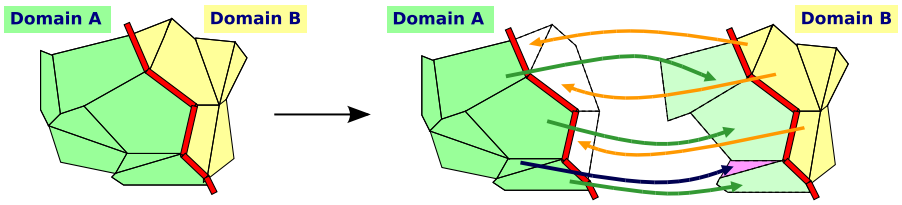

Figure 3: Left: Sketch of a domain partitioning; Right: Corresponding ghost cells.

competitive when using small cell counts per thread, and for a given number of total threads, memory consumption is usually reduced using several OpenMP threads per MPI ranks, possibly allowing runs on a higher number of threads when available memory per thread is limited.

3.2 CDO parallelisation approach

Layering the parallelism of new schemes on the current framework allows leveraging current partitioning, pre- and post-processing, and future *incremental approaches* combining CDO schemes with the legacy discretisation.

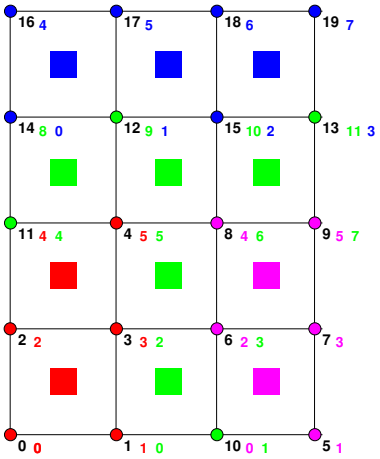

Figure 4: Example of 4x3 mesh partitioned into 4 ranks. The first rank is colored in red, the second rank in green, the third rank in magenta and the fourth in blue. The local numbering is given in script size with the color related to the rank it belongs to. Each mesh vertex is assigned to one and only one rank. The color of each mesh vertex translates this assignment. Then, the global numbering on mesh vertices results from a scan operation and is depicted in black and in normal size.

CDO schemes may be vertex- or face-based. This paper focuses on the vertex-based approach. Hence, some elements may appear on *multiple ranks* at partition boundaries. Moreover, we consider it important to *maintain compatibility* with external linear algebra libraries such as PETSc or HYPRE

- To benefit from their ongoing optimisation;
- To access numerous solvers for research.

Such libraries usually assume (1) that each element is assigned to a single rank, (2) each rank is assigned a contiguous range of matrix rows. To maintain this compatibility, we need to ensure each element is assigned to a single rank, and build a global numbering such that a contiguous range of global numbers is assigned to each rank. This is done by defining specific local and global numberings (see Figure 4 for an example):

1. Define an *owner rank* for elements on partition boundaries; this can be determined in parallel, using a highest adjacent rank or balancing rule. The important point is that using a simple struc-

ture describing partition boundaries and neighbour rank adjacencies, the rule may be computed on all ranks sharing an element with no additional computation, with identical results;

2. Locally order elements owned by current rank first, others last;

3. Define global ids based on local id on owning rank and sum of counts on previous ranks (parallel scan);

4. Assign range sets to each rank based on this numbering.

We then define an algebraic matrix assembler for simple and flexible construction of matrices based on local or global id couples. Compared to most existing libraries, the two stages relative to the structure determination and coefficient assignments are separated. This is because in most cases, a structure may be reused across time steps, while coefficients may vary from one time step to another depending on varying properties. Even when the mesh may change from one time step to another, the structure may usually be reused between systems related to different variables at each time step. When using such libraries, the structure may be used to provide precise initial dimensions, even if the library does excess work by recomputing the exact adjacencies.

3.2.1 Shared-memory parallelism

All the developments described here are done with shared memory parallelism in mind To this end, the following type of algorithms are dealt with: different threads read from different locations, but write only to one location whenever possible. This rationale is applied to the most CPU intensive part of the code which uses CDO schemes. Namely, the main part of the system building and some parts of the computation of mesh quantities and connectivities are performed looping on a range of cells for each thread or task. Algorithms rely on the usage of temporary and local arrays owned by each thread and by using specific structures also owned by each thread storing a cellwise view of the mesh and of the linear system.

For the assembly stage itself, scattering matrix coefficients from cells to vertices or faces needs to be thread-safe. Several strategies are possible: (1) use atomic sums or critical sections or (2) use renumbering/coloring.

The renumbering/coloring approach could possibly provide higher performance, but requires mostly static scheduling in the case of loop-level parallelism (i.e. is not compatible with work-stealing type schedulers). Some measure of dynamic scheduling could be used with a task-based approach, assigning subdomains to each task, and given a sufficiently small subdomain size so that there are many more tasks than threads. Currently, the first strategy is used based on critical sections. For cases where a static scheduling is used, standard memory accesses for elements which ar not shared could be used, and atomic accesses for those found at subdomain boundaries.

Threading write access conflicts could also be avoided by looping so as to always gather values rather than using scatter-gather patterns, but this would require storing unassembled values, with much higher memory and bandwidth usage.

Note also that elements assigned to a matrix assembly are simply appended with the matching row and column ids, and sorted based on increasing lexicographical row, column id values, so as to merge connectivity information and sum coefficient contributions. This is currently done using a heapsort algorithm, but adding values using linked lists of fixed-size bins could probably allow for better cache behavior and increased parallelism. Using the current API, only code internal to the assembler API would need to be further optimised.

4 Application to groundwater flows

4.1 The Richards equation

Following the notations of [30] and assuming a constant water density and no effect of the air phase, the Richards equation stating the mass conservation of water writes

$$\frac{\partial \theta(h)}{\partial t} - \mathrm{div}\left(k_r \underline{\underline{\kappa}}_s \cdot \underline{\mathrm{grad}}(H)\right) = 0, \tag{18}$$

where θ is the moisture content and k_r the relative permeability. The hydraulic head $H := h + \frac{\underline{g}}{|\underline{g}|} \cdot \underline{x}$ gathers the pressure head h and a gravitational potential $\frac{\underline{g}}{|\underline{g}|} \cdot \underline{x}$ ($\underline{g}$ denotes the gravitational vector). Pressure head and hydraulic head are equal if gravity effects are not accounted for. The pressure head $h = \frac{p}{\rho|\underline{g}|} + A$ where A is a constant such that $h = 0$ at the atmospheric pressure by convention. Eq. (18) is re-written in order to make appear only one variable H. This yields:

$$F\frac{\partial H}{\partial t} - \mathrm{div}\left(k_r \underline{\underline{\kappa}}_s \cdot \underline{\mathrm{grad}}(H)\right) = 0, \tag{19}$$

where $F = \frac{\partial \theta}{\partial h}$ is the water capacity. The Darcy flux is defined as $\underline{q}_D := -k_r \underline{\underline{\kappa}}_s \cdot \underline{\mathrm{grad}}(H)$. Non-linearities in (19) may arise from the definition of k_r and the law linking θ to h.

4.2 CDO scheme for the Richards equation

The DoFs attached to the discretisation of (19) corresponds to a discrete hydraulic head and are denoted by $\mathbf{H} \in \mathcal{V}$. The soil properties k_r, $\underline{\underline{\kappa}}_s$ and F are all approximated by a constant value inside each primal mesh cell. This value is computed at the cell barycentre. The semi-discrete counterpart of (19) writes as follows:

$$\frac{d}{dt}(\mathsf{H}_F^{\mathcal{V}\tilde{\mathcal{C}}}(\mathbf{H})) - \widetilde{\mathsf{DIV}} \cdot \mathsf{H}_{k_r\kappa}^{\varepsilon\tilde{\mathcal{F}}} \cdot \mathsf{GRAD}(\mathbf{H}) = \mathsf{B}, \tag{20}$$

where $\mathsf{B} \in \widetilde{\mathcal{C}}$ takes into account boundary conditions. The discrete pressure head is simply recovered by setting $\mathbf{h} := \mathbf{H} - \mathsf{R}_{\mathcal{V}}(\frac{\underline{g}}{|\underline{g}|} \cdot \underline{x}) \in \mathcal{V}$. Using an implicit Euler time scheme and approximating properties with the discrete pressure head at the previous time step yields:

$$\left(\mathsf{H}_{F^n}^{\mathcal{V}\tilde{\mathcal{C}}} - \widetilde{\mathsf{DIV}} \cdot \mathsf{H}_{k_r\kappa^n}^{\varepsilon\tilde{\mathcal{F}}} \cdot \mathsf{GRAD}\right)\mathbf{H}^{n+1} = \quad \mathsf{B} + \mathsf{H}_{F^n}^{\mathcal{V}\tilde{\mathcal{C}}}(\mathbf{H}^n) \tag{21}$$

The algebraic realisation of (21) is a symmetric positive definite linear system. Thus, a preconditionned conjuguate gradient algorithm can be used efficiently on this system.

Besides, the discrete Darcy flux $\mathbf{q} \in \widetilde{\mathcal{F}}$ writes

$$\mathbf{q} = -\mathsf{H}_{k_r\kappa_s}^{\varepsilon\tilde{\mathcal{F}}} \cdot \mathsf{GRAD}(\mathbf{H}). \tag{22}$$

This flux can be simply interpolated at cell centres as follows:

$$\underline{q}^{rec}(\underline{x}_c) := \frac{1}{|c|}\sum_{e\in \mathrm{E}} \mathbf{q}_e \underline{\tilde{f}}_c(e), \tag{23}$$

where $\underline{x}_c$ denotes the cell barycentre, $|c|$ the cell volume and $\underline{\tilde{f}}_c(e)$ the surface vector of the portion of dual face associated to an edge inside a primal mesh cell; see Figure 2 and [11, Prop. 5.24] for a detailed proof. This interpolation, relying on geometrical relations, is exact for constant vector fields as long as the primal mesh faces are planar.

The discretisation of the advective term devised in [13] relies on the expression of the Darcy flux (22) and yields either a centered approximation or an upwind approximation robust with respect to a local Peclet number. In [12], it is rather the expression (23) which is used to design a more accurate discretisation of the advection term based on *continuous interior penalty* technique. In what follows, the advection term is discretised using a centered approximation and a conservative formulation.

5 Verification test case

5.1 Description of the TRACY test case

In order to assess the discretisation of the Richards equation with CDO schemes, a one-dimensional verification test case proposed by Tracy [30] is considered. This test case simulates a variably saturated flow inside a horizontal column of height L (*i.e.* no gravity effect). The exact solution of the Richards equation is:

$$H(z,t) = h(z,t) = h_r \left(1 - \frac{\left(\frac{z-L}{L}\right)^2}{6 - \frac{5t}{T}} \right) \tag{24}$$

with the moisture content and the relative permeability defined as follows:

$$\theta = \theta_r + \frac{\theta_s - \theta_r}{h_s - h_r} \cdot (h - h_r), \qquad k_r = \frac{h - h_r}{h_s - h_r}. \tag{25}$$

The various parameters are $L = 200\ m$, $h_s = 0\ m$, $h_r = -100\ m$, $\theta_r = 0.15$, $\theta_s = 0.45$ and $\kappa_s = 10\ m.day^{-1}$. The time step is set to $\Delta t = 4320\ s$ and the final time is equal to $T = 10$ days, which corresponds to the time to get a totally saturated soil. Dirichlet boundary conditions at the top and bottom of the column as well as the initial condition are set according to (24). Homogeneous Neumann boundary conditions are imposed on the remaining parts of the geometry; see Figure 5.

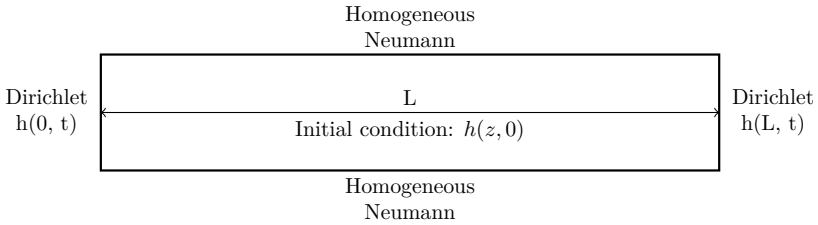

Figure 5: Sketch describing the geometry of the computational domain and the setting conditions for the TRACY test case.

5.2 Numerical results

Two polyhedral mesh sequences are considered: prismatic meshes with distorted hexagonal faces (denoted as PrG) and hexahedral meshes with hanging nodes (denoted as CB). Compared to the cubic domains depicted in Figure 1, the new domains used in this subsection only are obtained by stretching the vertex coordinates along the z-axis using a multiplification coefficient of 200. As a consequence the mesh quality is set to a (very) low level in order to assess the robustness of the CDO schemes. Assuming that N time steps have been performed, the accuracy is evaluated by computing a space-time relative l_2 norm defined as follows:

$$\mathbf{Er}_2(\mathbf{h})^2 := \Delta t \sum_{i=0}^{N} \frac{\sum_{v \in \mathrm{V}} |\tilde{c}(v)| (\mathbf{h}^i - h(\underline{x}_v, i\Delta t))^2}{\sum_{v \in \mathrm{V}} |\tilde{c}(v)| h(\underline{x}_v, i\Delta t)^2} \tag{26}$$

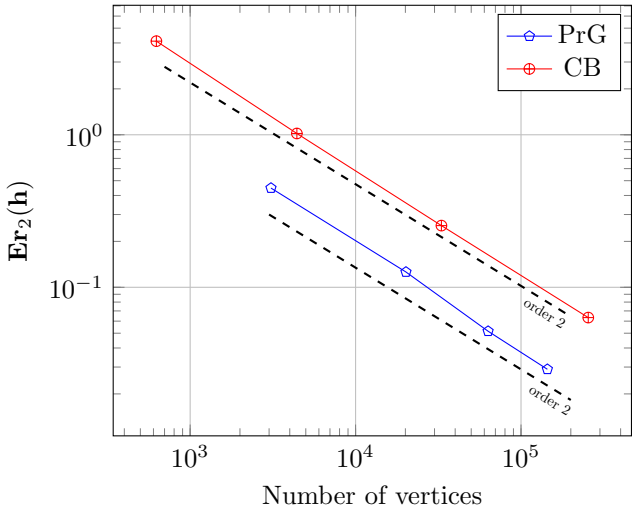

Figure 6: Space-time error on the pressure head with respect to the mesh refinement for the TRACY test case.

based on the difference between the results of the simulations and the analytical solution (24) provided by Tracy. Figure 6 shows the evolution of $\mathbf{Er}_2(\mathbf{h})$ as a function of the number of vertices for several PrG- (resp. CB-) type meshes. An expected second order convergence in space is observed for both types of mesh. Following [3], the rate of convergence R is computed as follows:

$$R := -3\frac{\dfrac{\mathbf{Er}_2[k]}{\mathbf{Er}_2[k-1]}}{\dfrac{\#\mathrm{V}[k]}{\#\mathrm{V}[k-1]}} \tag{27}$$

which corresponds to the ratio of two ratios, the numerator giving the ratio between the error for mesh $[k]$ and mesh $[k-1]$, and the denominator between the number of vertices #V for mesh $[k]$ and mesh $[k-1]$.

6 Validation of the method

6.1 Description of the HYDRUS test case

The HYDRUS test case simulates a variably saturated flow inside a vertical column where the gravity effect is taken into account. The case is one-dimensional and the soil is described using a Van Genuchten-Mualem model (cf. Eq. (28)).

$$S_e := \begin{cases} (1+\mid \alpha h \mid^n)^{-m} & \text{if } h < 0 \\ 1 & \text{if } h \geq 0 \end{cases} \tag{28a}$$

$$\theta := \theta_r + S_e \cdot (\theta_s - \theta_r) \tag{28b}$$

$$k_r := \begin{cases} k_s S_e^\tau \left(1 - \left(1 - S_e^{1/m}\right)^m\right)^2 & \text{if } h < 0 \\ k_s & \text{if } h \geq 0 \end{cases} \tag{28c}$$

The scale parameter α is set to $0.036\, m^{-1}$, the shape parameters $n = 1.56$ and $m = 1 - \dfrac{1}{n}$, the saturated conductivity $k_s = 0.3\, m.s^{-1}$, the residual moisture content $\theta_r = 0.078$, the saturated moisture content

$\theta_s = 0.3$ and the tortuosity parameter $\tau = 0.5$. The Darcy flux is then used to advect 3 pollutants. The transport of these pollutants is modelled according to the equation (29) presented for concentration c.

$$\frac{\partial(\theta + \rho K_d)c}{\partial t} + \mathrm{div}(\underline{q}_D\, c) - \mathrm{div}(\underline{\underline{D}}\,\underline{\mathrm{grad}}\, c) + \lambda(\theta + \rho K_d)c = 0 \tag{29}$$

where ρ represents the bulk density of the soil, K_d is the pollutant distribution coefficient, λ is the first-order decay coefficient and $\underline{\underline{D}}$ is the dispersion tensor defined in (30).

$$D_{ij} := \alpha_t |\underline{q}_D| \delta_{ij} + (\alpha_l - \alpha_t)\frac{q_{D,i} q_{D,j}}{|\underline{q}_D|} + d_m \delta_{ij} \tag{30}$$

where α_l (resp. α_t) is the longitudinal (resp. transversal) dispersivity, d_m is the water molecular diffusivity and $\underline{\underline{\delta}}$ is the identity tensor. In the current test case, $\rho = 1.5\, g.m^{-3}$, $\alpha_l = 1\, m$, $\alpha_t = 0\, m$ and $d_m = 0\, m^2.s^{-1}$. Table 1 gathers the values used for modelling specifically each pollutant.

Pollutant	$K_d\,[m^3.g^{-1}]$	$\lambda\,[s^{-1}]$
1^{st} tracer	0	0
2^{nd} tracer	0.1	0
3^{rd} tracer	0.1	0.01

Table 1: Parameters for the three pollutants used in the HYDRUS test case.

The same domain as for the TRACY test case (cf. Section 5) is considered. Homogeneous Dirichlet boundary conditions on the pressure head are imposed at the top and bottom of the column. The column is initially free of pollutants and the prescribed pressure head is $h(z, t = 0) = -\dfrac{30}{L}z$. Pollutants are injected inside the column from the top at a constant concentration equal to $1\, g.m^{-3}$ and a free exit is set at the bottom of the column. Homogeneous Neumann boundary conditions are set for the remaining parts of the domain. The time step is piecewise constant in order to manage the stronger non-linearities at the beginnnig of the computation so that $\Delta t = 0.025\, s$ if $t < 40\, s$ then $\Delta t = 0.25\, s$. 2160 time iterations are performed to reach the final simulation time equal to $180\, s$.

6.2 Physical results

Two hundred cubic PrG40 (resp. CB32) meshes are copied, shifted and glued together to generate the $1\ m \times 1\ m \times 200\ m$ domain. They are made of $13,448,000$ (resp. $29,491,200$) cells and $28,163,520$ (resp. $50,383,873$) vertices.

As no analytical solution exists either for the pressure head or the pollutant concentrations, comparison with the numerical results given by the HYDRUS [31] code are carried out.
Two representative time steps are presented, *i.e.* at $T = 10\ s$ and $T = 20\ s$. Figures 7 and 8 depict the pressure evolution as a function of the height of the domain and very good agreement with the reference data is observed. Figures 9 and 10 present the evolution of all 3 scalars as a function of the height. Again, the simulations match the data of the reference, showing that the first scalar propagate faster than the 2 others.

6.3 Performance analysis

The performance analysis is carried out for 2 types of architecture, the former using Intel IvyBridge CPUs only (ARCHER [32]) and the latter Intel Xeon Phi Knights Landing (ARCHER_KNL [32] and FRIOUL [33]). The meshes used in this section are all PrG-based.

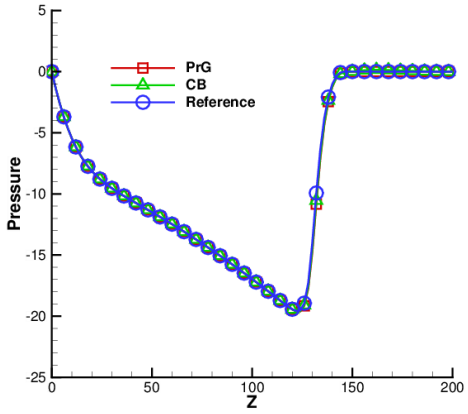

Figure 7: Evolution of the pressure as a function of the height at $T = 10\ s$.

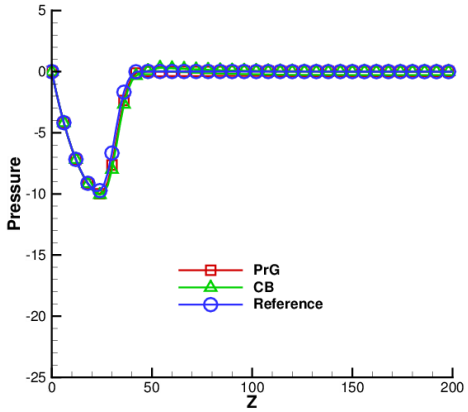

Figure 8: Evolution of the pressure as a function of the height at $T = 20\ s$.

6.3.1 Tests on CPUs using MPI only

The tests at scale are performed on a Cray X30 which consists of $4,920$ compute nodes connected together by the Aries interconnect. Each compute node contains two 2.7 GHz, 12-core E5-2697 v2 (Ivy Bridge) series processors. Within the node, the two processors are connected by two QuickPath Interconnect (QPI) links. Standard compute nodes on ARCHER have 64 GB of memory shared between the two processors. The memory is arranged in a non-uniform access (NUMA) form: each 12-core processor is a single NUMA region with local memory of 32 GB. Figure 11 shows the CPU time per time step as a function of the number of MPI tasks, when running 10 time steps of a 1.7 B cell simulation for the HYDRUS case. The mesh has been obtained by joining together $25,600$ shifted copies of the orginal PrG40 mesh which is made of $67,240$ prismatic cells. Nearly perfect scaling is observed up to $98,304$ cores, which represents about 89% of the machine.

6.3.2 Tests on KNLs using hybrid MPI-OpenMP

The KNL processor constitutes the second generation of Intel's Xeon Phi Many Integrated Core (MIC) architecture. It is entirely self-hosted. The ARCHER KNL (resp. FRIOUL) system is composed of

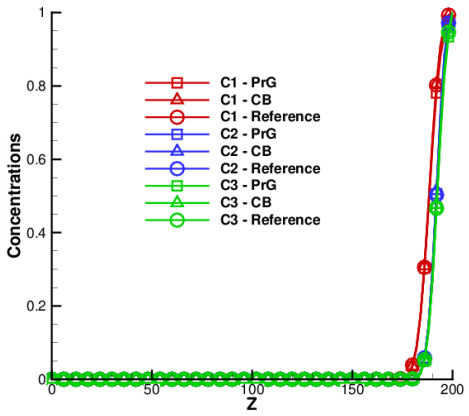

Figure 9: Evolution of the 3 passive scalars as a function of the height at $T = 10\ s$.

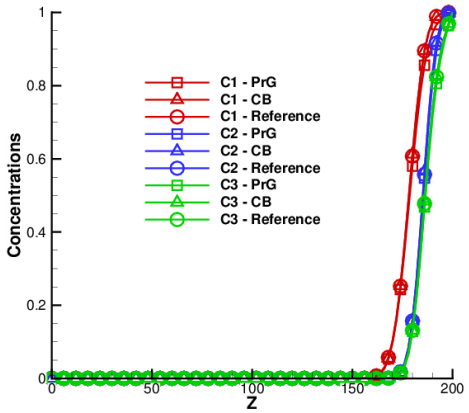

Figure 10: Evolution of the 3 passive scalars as a function of the height at $T = 20\ s$.

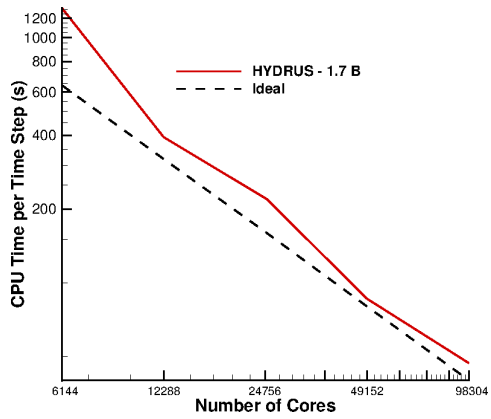

Figure 11: Performance of the vertex-based CDO scheme applied to a 1.7 B cell mesh using MPI only.

12 (resp. 48) compute nodes, each with a 64-core (resp. 68-core) KNL processor (model 7210 (resp. model 7250)) running at 1.30GHz (resp. 1.40GHz) and with each physical core capable of running 4 concurrent hyperthreads. Each compute node has 96GB (resp. 256GB) of system memory (DDR), and in addition to this each KNL processor comes with 16GB of on-chip memory (MCDRAM) which can be used in different modes. For ARCHER, the compute nodes are connected by a high-performance Cray Aries interconnect similar to that used in the main ARCHER machine, whether for FRIOUL, they are connected by Mellanox Infiniband. All the simulations run on both machines are carried out in cache mode, with 10 (resp. 24) of the 12 (resp. 48) for ARCHER (resp. for FRIOUL). In this case all 16 GB on-chip MCDRAM memory is automatically used to cache accesses to system DDR memory.

The mesh corresponds to the PrG-based mesh used in Section 6.2. Figures 12 and 13 show the

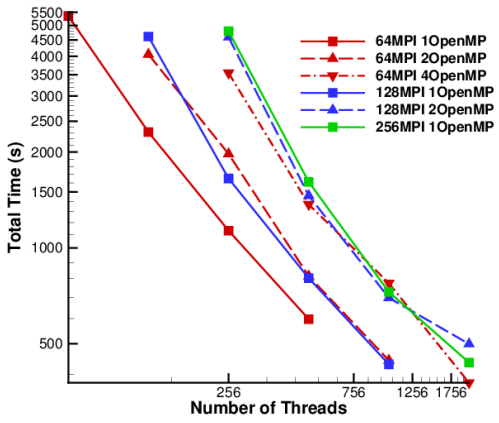

Figure 12: Evolution of the total time as a function of the number of threads on the KNLs with 64 cores. The square (resp. upper triangle/lower triangle symbols are used for 1 (resp. 2/4) thread(s).

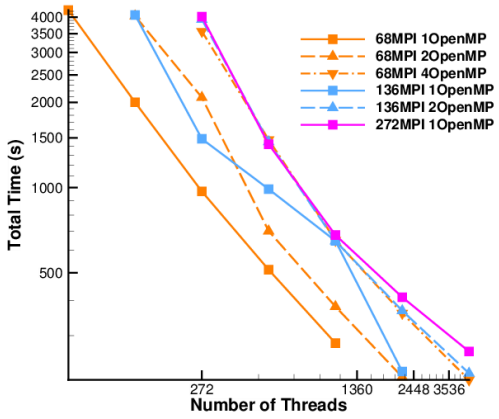

Figure 13: Evolution of the total time as a function of the number of threads on the KNLs with 68 cores. The square (resp. upper triangle/lower triangle symbols are used for 1 (resp. 2/4) thread(s).

evolution of the time to solution as a function of the number of threads, using either the KNL nodes with 64 physical cores or the KNL nodes with 68 physical cores. For both architectures, using 64 (resp. 68) MPI tasks and not taking advantage of hyperthreading is the worst case scenario. Conversely, using

4 OpenMP threads combined with 64 (resp. 68) MPI tasks per node shows the best performance on both architectures. Figure 14 presents a comparison of the performance of the code using 64 (resp. 68) MPI tasks and 4 OpenMP threads. The better timings are observed on the 68 cores per node machine, where more MPI tasks are dealt with at a higher frequency. Finally, Figures 15 and 16 plot the evolution as a function of the number of threads, of all the operations required for the code to reach solution, for both architectures and using 64 (resp. 68) MPI tasks and 4 OpenMP threads. All the operations scale well, apart some specially-dedicated post-processing, and also the matrix assembly stage which curve tails off, when increasing the number of threads. However this time to assemble the matrices is very small compare to the time to solve the equations.

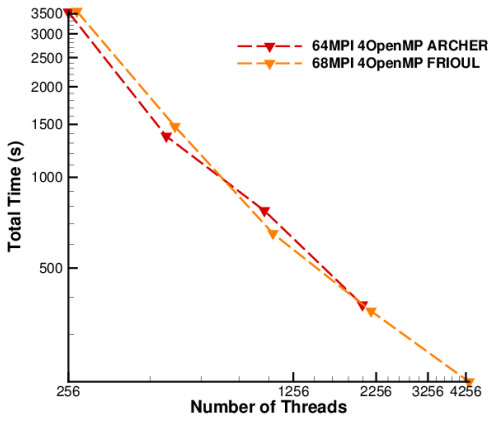

Figure 14: Comparison of the evolution of the total time on the KNLs with 64 cores and the KNLs with 68 cores. These two simulations are performed using 4 OpenMP threads.

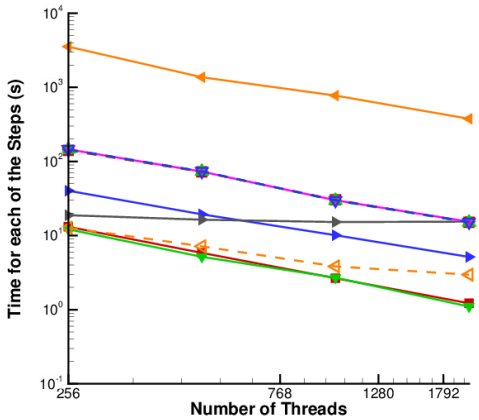

Figure 15: Case using 64 MPI tasks and 4 OpenMP threads. Evolution as a function of the number of threads of all the steps required for a simulation.

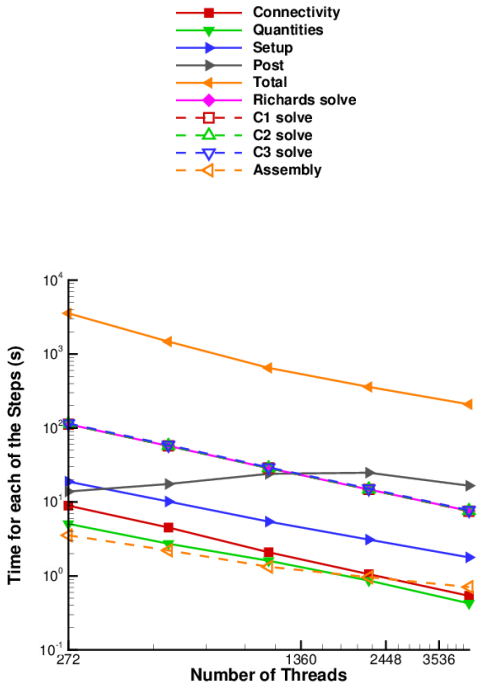

Figure 16: Case using 68 MPI tasks and 4 OpenMP threads. Evolution as a function of the number of threads of all the steps required for a simulation.

7 Conclusions - Future work

In this article, we have demonstrated that CDO vertex-based schemes are able to discretise accurately the transient Richards equation in unsaturated soil using polyhedral grids. The algorithm used to parallelise CDO schemes has demonstrated good performance at scale either on a traditionnal CPU architecture or on a modern KNL architecture using an hybrid MPI-OpenMP approach. This two-level approach appears to be the most effective.

A first prospect of this work is to further optimise the assembly process described in Section 3. Alternative prospects are an extension to higher order discretisation schemes such as the Hybrid High Order (HHO) approach [34] and the implementation of an in-house multigrid preconditionner for a conjuguate gradient (CG) solver taking benefit of the specifities of the CDO approach in order to speed-up the resolution of the Richards equation, for instance.

Acknowlegements

Some of the work has been carried out within the UK Turbulence Consortium (EP/L000261/1), PRACE 4iP and the SLA between STFC and EPSRC. The simulations on the KNLs with 64 cores per node were granted access on ARCHER by EPSRC and on the KNLs with 68 cores per node on FRIOUL by CINES and GENCI. The authors would also like to thank Huiling Zhan from the University of Manchester and CAAA for generating some of the meshes. An award of computer time was provided by the Innovative and Novel Computational Impact on Theory and Experiment (INCITE) program. This research used resources of the Argonne Leadership Computing Facility, which is a DOE Office of Science User Facility supported under Contract DE-AC02-06CH11357.

References

[1] R. Eymard, T. Gallouët, R. Herbin, Finite volume methods, in: Handbook of numerical analysis, Vol. VII, Handb. Numer. Anal., VII, North-Holland, 2000, pp. 713–1020.

[2] A. Ern, J.-L. Guermond, Theory and Pratice of Finite Elements, Vol. 159 of Applied Mathematical Sciences, Springer, 2004.

[3] R. Eymard, G. Henry, R. Herbin, F. Hubert, R. Klöfkorn, G. Manzini, 3D benchmark on discretization schemes for anisotropic diffusion problems on general grids, in: Finite Volumes for Complex Applications VI - Problems & Perspectives, Vol. 2, Springer, 2011, pp. 95–130.

[4] Y. Fournier, J. Bonelle, P. Vezolle, C. Moulinec, A. Sunderland, "an automatic joining mesh approach for computational fluid dynamics to reach a billion cell simulations, in: P. Iványi, B. Topping (Eds.), Proceedings of the Second International Conference on Parallel, Distributed, Grid and Cloud Computing for Engineering, Civil-Comp Press, 2011, paper 63. doi:10.4203/ccp.95.63.

[5] F. Archambeau, N. Mechitoua, M. Sakiz, Code_saturne: A Finite Volume code for the computation of turbulent incompressible flows - industrial applications, International Journal on Finite Volumes 1 (1).

[6] C. Moulinec, P. Wesseling, Colocated schemes for the incompressible NavierStokes equations on non-smooth grids for two-dimensional problems, International Journal for Numerical Methods in Fluids 32 (3) (2000) 349–364.

[7] M. C. Melaaen, Calculation of fluid flows with staggered and nonstaggered curvilinear nonorthogonal grids-the theory, Numerical Heat Transfer, Part B: Fundamentals 21 (1) (1992) 1–19. doi:10.1080/10407799208944919.

[8] M. C. Melaaen, Calculation of fluid flows with staggered and nonstaggered curvilinear nonorthogonal grids-a comparison, Numerical Heat Transfer, Part B: Fundamentals 21 (1) (1992) 21–39. doi:10.1080/10407799208944920.

[9] J. Bonelle, A. Ern, Analysis of Compatible Discrete Operator schemes for the Stokes equations on polyhedral meshes, IMA J. Numer. Anal.doi:10.1093/imanum/dru051.

[10] J. Bonelle, A. Ern, Analysis of Compatible Discrete Operator Schemes for Elliptic Problems on Polyhedral Meshes, ESAIM: Mathematical Modelling and Numerical Analysis 48 (2) (2014) 553–581. doi:10.1051/m2an/2013104.

[11] J. Bonelle, Compatible Discrete Operator schemes on polyhedral meshes for elliptic and Stokes equations, Ph.D. thesis, Université Paris-Est (2014). doi:10.13140/2.1.4866.0805.

[12] P. Cantin, J. Bonelle, E. Burman, A. Ern, A vertex-based scheme on polyhedral meshes for advection-reaction equations with sub-mesh stabilization, Computers & Mathematics with Applications 72 (9) (2016) 2057 – 2071. doi:http://dx.doi.org/10.1016/j.camwa.2016.07.038.

[13] P. Cantin, A. Ern, Vertex-based compatible discrete operator schemes on polyhedral meshes for advection-diffusion equations, Comput. Meth. Appl. Math. 16.

[14] F. Brezzi, A. Buffa, K. Lipnikov, Mimetic Finite Difference for elliptic problem, ESAIM: Mathematical Modelling and Numerical Analysis 43 (2) (2009) 277–295. doi:10.1051/m2an:2008046.

[15] F. Brezzi, K. Lipnikov, M. Shashkov, Convergence of the Mimetic Finite Difference method for diffusion problems on polyhedral meshes, SIAM J. Numer. Anal 43 (5) (2005) 1872–1896. doi:10.1137/040613950.

[16] L. Codecasa, F. Trevisan, Constitutive equations for discrete electromagnetic problems over polyhedral grids, Journal of Computational Physics 225 (2) (2007) 1894–1918. doi:10.1016/j.jcp.2007.02.032.

[17] R. Eymard, C. Guichard, R. Herbin, Small stencil 3D schemes for diffusive flows in porious media, ESAIM: Mathematical Modelling and Numerical Analysis 46 (2012) 265–290.

[18] R. Eymard, T. Gallouët, R. Herbin, Discretization of heterogeneous and anisotropic diffusion problems on general nonconforming meshes SUSHI: a scheme using stabilization and hybrid interfaces, IMA Journal of Numerical Analysis 30 (4) (2010) 1009–1043.

[19] J. Droniou, R. Eymard, T. Gallouët, R. Herbin, Gradient Schemes: a generic framework for the discretisation of linear, nonlinear and nonlocal elliptic and parabolic equations, Mathematical Models and Methods in Applied Sciences 23 (13) (2013) 2395–2432.

[20] Y. Fournier, J. Bonelle, C. Moulinec, Z. Shang, A. G. Sunderland, J. C. Uribe, Optimizing Code_Saturne computations on Petascale systems, Computers & Fluids 45 (1) (2011) 103–108. doi:10.1016/j.compfluid.2011.01.028.

[21] A. Bossavit, Computational electromagnetism and geometry, J. Japan Soc. Appl. Electromagn. & Mech. 7-8 (1999-2000) 150–9 (no 1), 294–301 (no 2), 401–8 (no 3), 102–9 (no 4), 203–9 (no 5), 372–7 (no 6).

[22] P. Bochev, J. M. Hyman, Principles of mimetic discretizations of differential operators, in: D. Arnold, P. Bochev, R. Lehoucq, R. A. Nicolaides, M. Shashkov (Eds.), Compatible Spatial Discretization, Vol. 142 of The IMA Volumes in mathematics and its applications, Springer, 2005, pp. 89–120.

[23] M. Desbrun, E. Kanso, Y. Tong, Discrete differential forms for computational modeling, in: Oberwolfach Seminars, Springer Science – Business Media, 2008, pp. 287–324. doi:10.1007/978-3-7643-8621-4_16.

[24] G. Kron, A Set of Principles to Interconnect the Solutions of Physical Systems, J. Appl. Phys. 24 (1953) 965. doi:10.1063/1.1721447.

[25] F. Branin, The Algebraic-Topological Basis for Network Analogies and the Vector Calculus, in Symposium on generalized networks (12–14 April 1966) (1966).

[26] R. Hiptmair, Discrete Hodge operators: An algebraic perspective, Progress In Electromagnetics Research 32 (2001) 247–269.

[27] J. Bonelle, D. A. D. Pietro, A. Ern, Low-order reconstruction operators on polyhedral meshes: application to compatible discrete operator schemes, Computer Aided Geometric Design 3536 (2015) 27 – 41. doi:http://dx.doi.org/10.1016/j.cagd.2015.03.015.

[28] L. Codecasa, R. Specogna, F. Trevisan, A new set of basis functions for the Discrete Geometric Approach, Journal of Computational Physics 229 (19) (2010) 7401–7410. doi:10.1016/j.jcp.2010.06.023.

[29] D. A. Di Pietro, S. Lemaire, An extension of the Crouzeix-Raviart space to general meshes with application to quasi-incompressible linear elasticity and Stokes flow, Math. Comp. 84 (291) (2015) 1–31.

[30] F. Tracy, 1-d, 2-d, and 3-d analytical solutions of unsaturated flow in groundwater, Journal of Hydrology 170 (1995) 199–214.

[31] J. Simunek, R. van Genuchten, M. Sejna, Development and applications of the HYDRUS and STANMOD software packages and related codes, Vadose Zone Journal 7 (2) (2008) 587–600. doi:10.2136/vzj2007.0077.

[32] http://www.archer.co.uk.

[33] https://www.cines.fr/le-supercalculateur-frioul.

[34] D. A. Di Pietro, A. Ern, 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) (2014) 461–472. doi:10.1515/cmam-2014-0018.