Journal of Computational Physics 355 (2018) 59–77

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

A finite-difference method for the variable coefficient Poisson equation on hierarchical Cartesian meshes

Alice Raeli [a,b,*], Michel Bergmann [a,b], Angelo Iollo [a,b]

[a] Institut de Mathématiques de Bordeaux, UMR 5251, Université de Bordeaux, 33405 Talence, France
[b] INRIA Bordeaux – Sud Ouest, MEMPHIS team-project, 33405 Talence, France

ARTICLE INFO

Article history:
Received 25 July 2017
Received in revised form 3 November 2017
Accepted 4 November 2017
Available online 10 November 2017

Keywords:
Finite difference method
Hierarchical Cartesian grid
Octree/quadtree
Variable coefficient Poisson equation

ABSTRACT

We consider problems governed by a linear elliptic equation with varying coefficients across internal interfaces. The solution and its normal derivative can undergo significant variations through these internal boundaries. We present a compact finite-difference scheme on a tree-based adaptive grid that can be efficiently solved using a natively parallel data structure. The main idea is to optimize the truncation error of the discretization scheme as a function of the local grid configuration to achieve second-order accuracy. Numerical illustrations are presented in two and three-dimensional configurations.

© 2017 Elsevier Inc. All rights reserved.

1. Introduction

Many applications like for example incompressible multi-phase fluid flows or heat conduction in non-homogeneous materials are characterized by strongly varying physical parameters across internal interfaces. In classical approaches these interfaces are treated like internal boundaries using interface fitted meshes, see for example [9] for a recent heat-conduction application. These methods are accurate and can lead to simple discretization schemes of the interface conditions. However, grid generation and handling can be costly and cumbersome when the interface geometry is evolving in time. Furthermore, solution in parallel typically requires time-dependent partitioning of the grid that induces additional computational costs.

In the present approach we use non-conforming hierarchical meshes to discretize the solution. The hierarchical nature of the grid makes mesh generation, adaptivity and partitioning very efficient and with a low-memory footprint. Since the grid is non-conforming to these internal boundaries, discontinuous coefficients are regularized across the interfaces. Accuracy is then recovered thanks to grid adaptivity.

Johansen and Colella [10] were among the first to propose adaptive block-cartesian meshes to solve the Poisson equation. They proposed a cell-centered second-order scheme based on local quadratic reconstructions. In the same spirit Howell and Bell [8] solved a Poisson problem within a projection method for incompressible viscous flows using a quadratic reconstruction ghost-cell approach to recover appropriate accuracy at the border between different blocks. A different setting was proposed by Popinet [17] to solve the incompressible Euler equations with octree grids. For the Poisson solver, the author proposes a cell-centered discretization scheme using all first neighbors in order to recover a second-order approximation as a function of the local grid configuration. Losasso et al. [11] proposed a Poisson solver on octree which converges to the actual solution with second-order accuracy. This approach allows a discretization matrix which is symmetric so that the con-

* Corresponding author.
E-mail address: alice.raeli@math.u-bordeaux.fr (A. Raeli).

https://doi.org/10.1016/j.jcp.2017.11.007
0021-9991/© 2017 Elsevier Inc. All rights reserved.

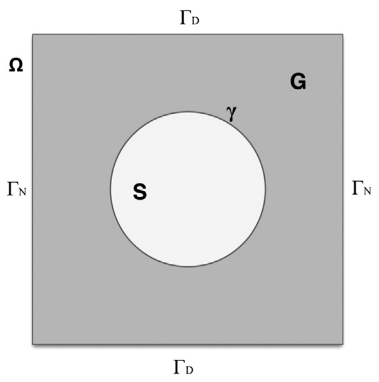

Fig. 1. Sketch of the domain under consideration.

vergence rate of the iterative solvers is significantly improved. Min et al. [13] recover a higher-order accuracy with a compact stencil at the price of moving from a cell-centered scheme to a vertex-centered scheme. Other examples of node-centered schemes with a direct application to level set methods are given in Losasso et al. [12] and Mirzadeh et al. [14]. Efficient parallel algorithms that can handle node-centered grids are presented in Burstedde et al. [3], whereas a finite-element application is shown in Bangerth et al. [2]. Finally, a Poisson solver based on a finite-volume approach with a least-square reconstruction of the fluxes at the octree level jumps is proposed by Olshanskii et al. [16]. More recently, a Voronoi Interface Method is presented by Guittet et al. [7] for general elliptic problems with subdomain discontinuities. In this approach additional degrees of freedom are placed close to the subdomain interface and a Voronoi partition centered at each of these points is used to discretize the equations in a finite volume approach. The solution obtained is second-order accurate.

In this work we present a cell-centered finite-difference scheme to solve a variable coefficient Poisson equation on quadtrees and octrees. The main idea of the method is to optimize the truncation error of the elliptic operator discretization as a function of the local grid configuration. The optimization problem solution is local, fast and it involves only the first neighbors in order to reduce communications in parallel. The scheme is second-order accurate when the coefficients vary smoothly through the domain. In the following we describe the method and assess its consistency, accuracy and extension to general hierarchical grids in two and three dimensions. Numerical illustrations are presented in two and three-dimensional configurations.

2. Problem definition

We consider a configuration representing an idealized composite medium. A domain Ω is subdivided in two parts, G and S. We suppose that different diffusion parameters characterize the two sub-domains (Fig. 1); we have $\Omega = G \cup S$ and γ the interface between both sub-domains. We distinguish Neumann Γ_N and Dirichlet Γ_D boundary conditions on the external boundary.

The variable coefficient Poisson problem we consider is modeled by:

$$-\mathrm{div}(\kappa(x)\nabla u(x)) = g(x) \qquad \text{in } G \cup S, \tag{1a}$$

$$\partial_n u(x) = 0 \qquad \text{on } \Gamma_N, \tag{1b}$$

$$u(x) = u_D(x) \qquad \text{on } \Gamma_D, \tag{1c}$$

$$[\kappa(x)\partial_n u(x)] = 0, \quad [u] = 0 \qquad \text{on } \gamma, \tag{1d}$$

where $x \in \mathbb{R}^n$ are the spatial coordinates and $\kappa(x)$ is piecewise continuous on each subdomain but possibly discontinuous across γ. In that case, the solution $u(x)$ is continuous all over the domain $G \cup S$ and the normal derivatives are discontinuous across γ.

In the following we will initially describe the data structure. Then we consider the Laplace operator and investigate its discretization, consistency and accuracy. We moreover detail how to accurately impose Dirichlet boundary conditions on unfitted grid boundaries via penalization and hierarchical grid refinement. The variable coefficient Poisson equation in the limit of discontinuous parameters is finally considered.

3. Hierarchical grid data structure

We consider a hierarchical data structure based on the principle of recursive decomposition of space. The decomposition is done into equal parts on each level. Each internal node has exactly four children (quadtree) for two-dimensional problems, and eight children (octree) for three-dimensional problems. The quadtree is defined in a square, the octree in a cube. For simplicity reasons, we describe the data structure in two dimensions (Fig. 2).

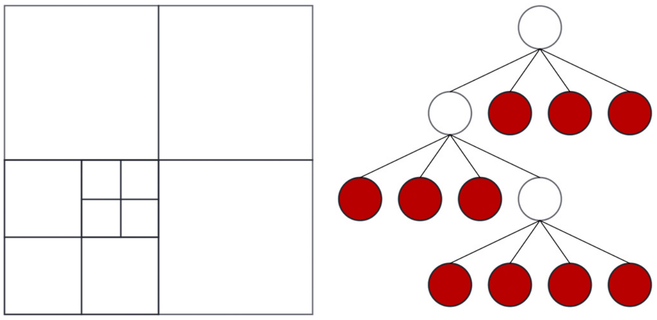

Fig. 2. A square decomposition and the corresponding quadtree.

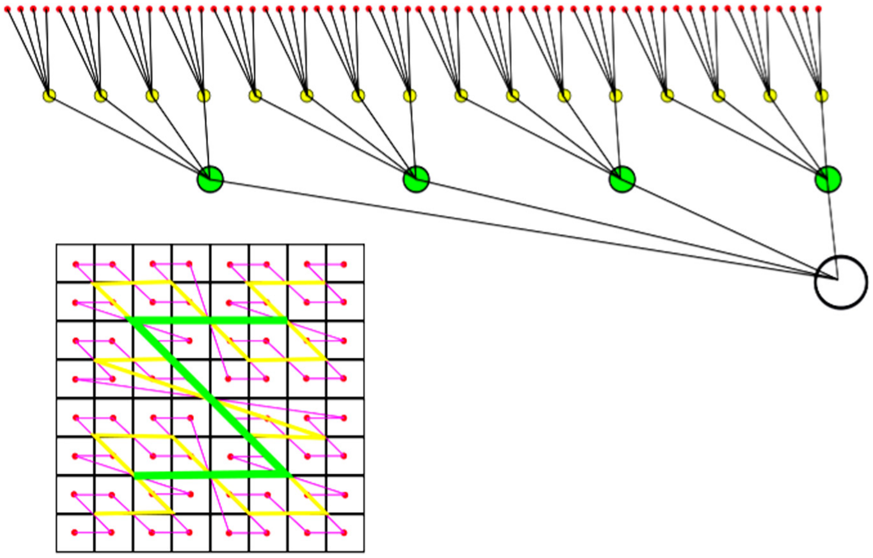

Fig. 3. Global nested Z-ordering example.

The data structure is based on a linear octree [6], therefore only the leafs of the tree structure (see Fig. 2) are stored. This linear data structure is easily dispatched to a distributed memory architecture. In order to limit parallel communications, we constrain the discretization scheme to include only the first layer of neighboring cells. We use a cell centered scheme because it is easier to handle, but the overall process can be applied to vertex center schemes. In what follows we will refer to each point in the space as a *cell* or *octant*. Each cell may be the *parent* of four (eight in 3D) *children*. The *root cell* is the base of the tree (often it represents the entire region before the discretization) and a *leaf* is a cell without any child. The *level* of a cell is defined by starting from zero for the root cell and by adding one every time a group of descendant children is appended. Each cell C has two kinds of *neighbors*: through faces following the axial directions and through corners following its diagonals directions.

Hierarchical grids are defined graded (or balanced) if the levels of all neighboring cells do not differ by more than one. This constraint has little impact on the flexibility of the discretization we propose but it may allow a gradual refinement by increments of two. In the following, we will mainly focus on graded grids but we will present typical results on non-graded grids in order to stress that the scheme can be applied without modifications.

The octree data structure is designed based on the following requirements: i) efficient access to neighboring cells; ii) efficient access to cell positions and their levels; iii) efficient access to stored data. To this end, we assign a Z-order index to each cell ([15], Fig. 3) thanks to the library PABLO.[1]

We classify the neighbors topology of an octant using a base-5 8-digits numerical key (resp. 26-digits for the 3D case). We define a function of the level: $[L] := L - nL$, with L the level of the actual octant and nL the level of the neighbor. The values attributed to the key elements are presented in Table 1. An example of this construction is given in Fig. 4.

This key has the following properties: it is bijective, it is easy to build and to interpret, and it is independent of the cell dimensions, i.e., the tree level. The use of this key can lead to a significant speed up in the grid pre-processing phase.

[1] http://www.optimad.it/products/bitpit/.

Table 1
Key values attribution.

Assigned value	Possible cases
0	∄ neighbor on this side
1	$[L]=0$
2	$[L]=1$
3	$[L]=-1$
4	$[L]=2$
5	$[L]=-2$

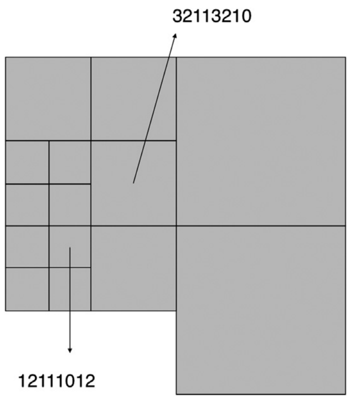

Fig. 4. Example of bijective key construction.

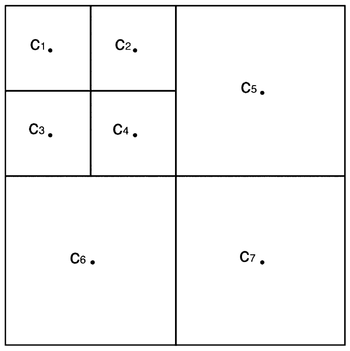

Fig. 5. Discretization grid of the Laplacian in c_4.

4. Cell-centered finite-difference discretization of the Laplacian

In this section we present the main idea to discretize the Laplacian. A similar approach is used for the gradients.

There are two natural choices to discretize differential operators on hierarchical grids: vertex-centered [13] or cell-centered [17]. Here we considerer a cell-centered scheme. In this case, thanks to the data structure we use, the neighbor configuration is more easily accessible compared to a vertex-centered scheme.

The main idea is to ensure consistency and second-order accuracy of the truncation error in the sense of finite differences as a function of the number of neighbors. Let us focus on a two-dimensional problem and let us consider the configuration in Fig. 5. As shown by Min et al. [13], if only face-adjacent cells are to be used to discretize the Laplace operator in c_4, then there is no locally consistent linear scheme in the sense of finite differences. Instead, we discretize the Laplace operator in c_4 using all the points belonging to the first layer of neighbors. This will allow us to obtain more degrees of freedom than sufficient constraints for consistency. Possibly, as a function of the number of available points and symmetries, we will also ensure sufficient conditions for second-order accuracy. To see this, let h be the side length of the cell c_4. To obtain a consistent scheme we must ensure that the discretization coefficients a_i, $1 \le i \le 7$ satisfy:

$$u_{xx}+u_{yy}=a_1u_1+a_2u_2+a_3u_3+a_4u_4+a_5u_5+a_6u_6+a_7u_7+O(h).$$

Using standard Taylor analysis, we expand for example the solution u_3 in c_3 with respect to u_4 in c_4 and get:

$$u_3=u_4-h\frac{\partial u_4}{\partial x}+\frac{h^2}{2}\frac{\partial^2 u_4}{\partial x^2}+O(h^3).$$

Similarly, all the other points are expanded with respect to c_4. A complete Taylor analysis on all the involved neighbors leads to the following linear system:

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & -h & 0 & -h & \frac{3h}{2} & -\frac{h}{2} & \frac{3h}{2} \\ 0 & h & h & 0 & \frac{h}{2} & -\frac{3h}{2} & -\frac{3h}{2} \\ 0 & \frac{h^2}{2} & 0 & \frac{h^2}{2} & \frac{9h^2}{8} & \frac{h^2}{8} & \frac{9h^2}{8} \\ 0 & -h^2 & 0 & 0 & \frac{3h^2}{4} & \frac{3h^2}{4} & -\frac{9h^2}{4} \\ 0 & \frac{h^2}{2} & \frac{h^2}{2} & 0 & \frac{h^2}{8} & \frac{9h^2}{8} & \frac{9h^2}{8} \end{pmatrix} \begin{pmatrix} a_4 \\ a_1 \\ a_2 \\ a_3 \\ a_5 \\ a_6 \\ a_7 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}.$$

In the example above we must determine seven discretization coefficients a_i, $1 \le i \le 7$ but we only have six constraints for consistency. The idea is to ensure consistency and, at the same time, to minimize the deviation from second-order accuracy as follows.

In the general case, when the number of constraints is m, we solve the constrained minimization problem by defining an appropriate Lagrangian function. Let $\lambda \in \mathbb{R}^m$ a vector of Lagrange multipliers, $a \in \mathbb{R}^n$ the discretization coefficient vector of size n (the size of the stencil), $M \in \mathcal{M}^{m,n}(\mathbb{R})$ the constraint matrix, $f \in \mathbb{R}^m$ the right hand side vector corresponding to the imposed constraints and $F(a)$ a convex cost function from $\mathbb{R}^n$ to $\mathbb{R}$. We define a Lagrangian function $\mathcal{L}(a,\lambda)$: $\mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ as follows

$$\mathcal{L}(a,\lambda) = F(a) - \lambda^T(Ma - f), \tag{2}$$

and compute the stationary point of this function with respect to (a,λ):

$$\begin{cases} \frac{\partial \mathcal{L}(a,\lambda)}{\partial a} = & 0, \\ \frac{\partial \mathcal{L}(a,\lambda)}{\partial \lambda} = & 0. \end{cases} \Leftrightarrow \begin{cases} \frac{\partial F}{\partial a} - M^T\lambda & = 0, \\ Ma & = f. \end{cases}$$

Let $B \in \mathcal{M}^{6,n}$ the sub-matrix corresponding to the consistency constraints, $C \in \mathcal{M}^{4,n}$ the sub-matrix relative to the second-order constraints, $\alpha \in [0,1]$ and let $h = 1$. The discretization coefficients are then rescaled dividing by the appropriate value of the cell side. We distinguish two cases:

- $n \le 10$: $M = B$ and we take $F(a) = 1/2a^T\left((1-\alpha)C^TC + \alpha I\right)a$ and the local system to be solved is

$$\begin{pmatrix} ((1-\alpha)C^TC + \alpha I) & -B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} a \\ \lambda \end{pmatrix} = \begin{pmatrix} 0 \\ f \end{pmatrix}.$$

  This choice of the convex function $F(a)$ is such that the discretization coefficients minimize the second-order truncation error encoded in matrix C and their norm is penalized by coefficient α. We have chosen a small value of α that results in a stable matrix to invert and that introduces the minimal amount of regularization. We took $\alpha = 0.01$ for all the numerical illustrations in the following. The coefficients a always satisfy 6 consistency constraints.
- $n > 10$: $M = \binom{B}{C}$ $F(a) = 1/2a^T a$ $m = 10$ $I \in \mathcal{M}_{n,n}$

$$\begin{pmatrix} I & -M^T \\ M & 0 \end{pmatrix} \begin{pmatrix} a \\ \lambda \end{pmatrix} = \begin{pmatrix} 0 \\ f \end{pmatrix}.$$

  The coefficients satisfy 10 second-order accuracy constraints while their norm is minimized.

This approach is independent of the specific grid configuration and can be applied to either graded or non-graded grids. Although we use a cell-centered stencil, this method can in principle be applied to vertex-centered stencils. We remark that the minimal number of available points including only the first neighbors in 2D is 7 if the grid is graded and 6 if the grid is non-graded. Therefore, the discretization will always be at least consistent.

Remark 4.1 *(Uniform mesh).* The discretization weights (Fig. 6) are $-1.\bar{3}$ for the red point (center of the configuration), $-0.\bar{3}$ for the face adjacent cells marked in green and $0.\bar{6}$ for the corner blue points. The resulting truncation error weights ensures order two convergence.

Remark 4.2 *(Three-dimensional extension).* The neighbors are found through faces, edges and vertexes. The consistency constraints are 10, the number of equations to obtain second order accuracy is 20. For either graded or non-graded grids, the scheme will be at least consistent since with graded grids we have at least 15 available points including only first neighbors and with non-graded 11. Beyond 30 available points, in order to limit the size of the stencil, we consider the minimum number of all possible neighbors satisfying consistency and second-order accuracy. Hence, where possible, neighbors through edges are not considered and the stencil takes into account only neighbors through faces and vertices.

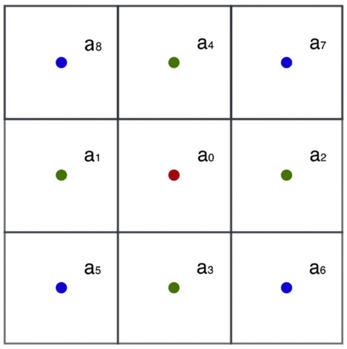

Fig. 6. Uniform mesh configuration. The weights are enumerated following the Z-order. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

5. Internal boundaries

5.1. Dirichlet boundary conditions and penalization

Internal boundaries with Dirichlet type conditions are modeled by a penalty term [1]. Let χ_c be the characteristic function of a given domain c, e.g., the circle in Fig. 10. Let us consider the equation

$$\Delta u_\varepsilon = g - \frac{\chi_c}{\varepsilon}(u_\varepsilon - u_0). \tag{3}$$

We set $u_\varepsilon = u + \varepsilon \tilde{u}$ to derive the equations satisfied by u and $\tilde{u}$. By identifying the terms of the same order in ε we have $\chi_c(u - u_0) = 0$ and $\Delta u = \chi_c \tilde{u}$. This formally implies that $u = u_0$ in the circle and $\Delta u = g$ outside. Further analysis [4] shows that $\|u_\varepsilon - u\|_2 = O(\sqrt{\varepsilon})$.

The numerical discretization of the penalized model on an unfitted boundary will introduce an additional discretization error of order h, so that the scheme will be only first order accurate. Second-order penalization can be obtained by extrapolation as shown in [5].

5.2. Diffusion coefficient discontinuity

In many applications the diffusion coefficient $\kappa(x)$ can abruptly vary across an interface between two positive constants α and β. However, the normal fluxes are continuous at the interface. We model these problems by the regularized diffusion function

$$\kappa(x) = \alpha + (\beta - \alpha)\left(\frac{\tanh(\sigma \cdot \Phi(x)) + 1}{2}\right), \tag{4}$$

where $\Phi(x)$ is the signed distance function with respect to the interface of discontinuity and σ the regularization parameter. The signed distance function is obtained by solving $|\nabla\Phi(x)| = 1$ with $\Phi(x) = 0$ on the interface. Due to regularization, we expect a first-order convergence near the interface (in the infinite norm).

6. Results and discussion

In what follows, the linear systems are solved using the PETSc library.[2] In most of the following cases, we used a block Jacobi preconditioning (BJACOBI) on a global flexible GMRES. In two dimensions preconditioning was not strictly necessary. In three dimensions, besides the Jacobi preconditioning, sub-preconditioners of type ILU were employed.

6.1. Consistency

We consider here the case $\kappa(x) = 1$. The domain is a $[0, 1] \times [0, 1]$ square and the grid studied is a bi-periodic lattice obtained by initially repeating the elementary configuration of Fig. 5 as presented in Fig. 7. It has been shown [13] that the numerical scheme is inconsistent for this kind of grids if we only use the face neighbors. We thus use all the neighbors including the edge neighbors. The problem to be solved is $\Delta u(x) = f(x)$. We considered a test case with the exact solution $u_e(x_1, x_2) = \sin((x_1 - 0.5)^2 + (x_2 - 0.5)^2)$. The convergence Table 2 is obtained by subsequently subdividing each cell. A second-order accuracy is obtained for both L^2 and L^∞ norms. An example of the error distribution is given in Fig. 8.

[2] https://www.mcs.anl.gov/petsc/.

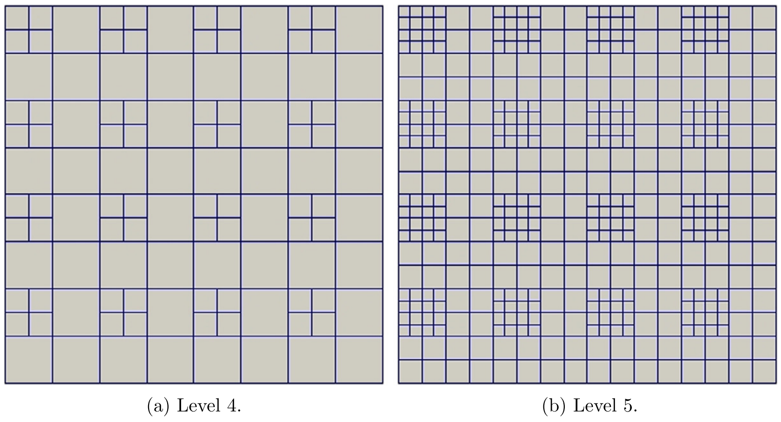

Fig. 7. Example of the mesh configuration for level 4 and 5.

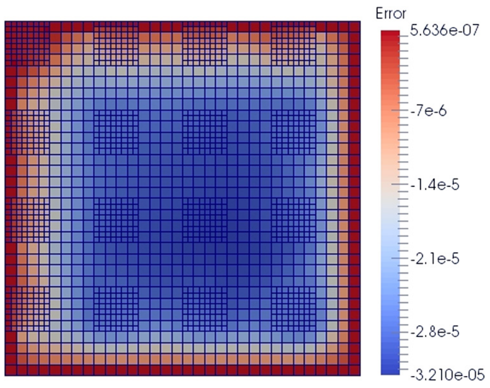

Fig. 8. Example of error distribution on a grid corresponding to level 6.

Table 2
Error norms and order of the scheme.

Tree level	L^∞	L^2	Order L^∞	Order L^2
4	5.02265e−04	2.49066e−04		
5	9.92104e−05	4.96958e−05	2.5315	2.506
6	2.23894e−05	1.13086e−05	2.2155	2.19725
7	5.36436e−06	2.72476e−06	2.087	2.075
8	1.31696e−06	6.70944e−07	2.0365	2.0305

Table 3
Convergence order for the random mesh generation. The first line involves 787 degrees of freedom, the last one 3223552.

Tree level	L^∞	L^2	Order L^∞	Order L^2
6	1.18078e−04	7.37138e−05	–	–
7	2.93059e−05	1.94914e−05	2.010	1.929
8	6.40356e−06	4.3646e−06	2.194	2.159
9	1.52669e−06	1.02794e−06	2.068	2.086
10	3.72553e−07	2.48591e−07	2.035	2.048
11	9.20017e−08	6.10625e−08	2.018	2.025
12	2.28589e−08	1.51279e−08	2.01	2.013

6.2. Random mesh generation

To check the robustness of the scheme ruling out any possibility of error cancellation due to the regularity of the mesh, we applied a random algorithm to generate the grid (Fig. 9). The mesh is then subsequently refined. The convergence order of the method is given in Table 3.

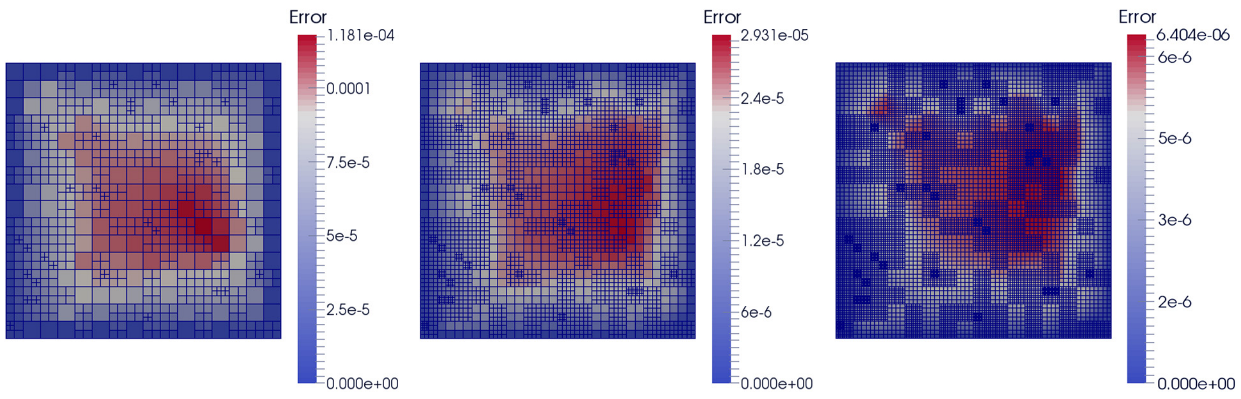

Fig. 9. Error distribution for random mesh generation. Levels 6, 7 and 8 respectively.

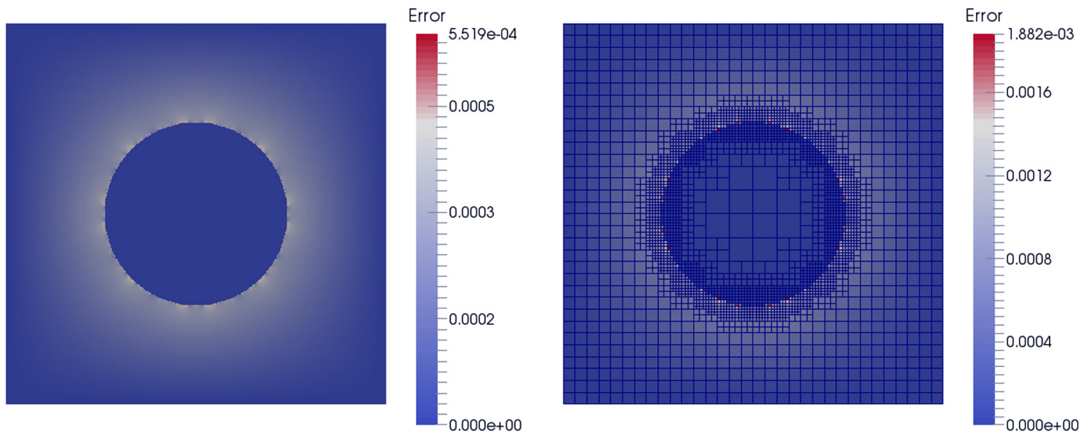

Fig. 10. Error distribution for the penalized cylinder with a maximum tree level 9 and 7 respectively.

Table 4
Numerical results for the penalized cylinder.

Tree level	L^∞	L^2	Order L^∞	Order L^2
5	1.39209e−02	3.21637e−03		
6	7.16906e−03	1.50085e−03	0.971	1.071
7	3.84769e−03	7.15689e−04	0.932	1.048
8	1.95191e−03	3.68745e−04	0.986	0.970
9	9.72571e−04	1.56265e−04	1.003	1.18
10	4.88266e−04	7.96255e−05	0.996	0.981

6.3. Dirichlet boundary conditions

Boundary conditions on the interior subdomain are imposed by a penalty term, see equation (3). We consider the same domain and exact solution $u_e(x_1, x_2) = \sin((x_1 - 0.5)^2 + (x_2 - 0.5)^2)$ as in the previous section. The solution is penalized on all the mesh nodes lying inside a centered cylinder with radius equal to 0.25 with the value at the cylinder boundary. Equation (3) is thus solved with $\varepsilon = 10^{-11}$ and $u_0 = \sin((0.25)^2 + (0.25)^2)$. The grid is subsequently refined in a graded way according to the distance function Φ to the cylinder in a layer $|\Phi(x)| \le 0.2$. The error distribution in the computational domain for tree level 7 is presented In Fig. 10. Numerical results are reported in Table 4. As expected, since the internal Dirichlet boundary conditions are not exactly imposed on the cylinder boundary, the maximum errors are observed near the cylinder boundary and the numerical solution is only first order accurate.

6.4. Uniform refinement and AMR for a multiscale problem

We investigate an idealized multiscale problem with the same domain and exact solution as in previous sections, but now we consider a penalized centered cylinder with radius equal to 0.01. Equation (3) is now solved with $u_0 = \sin((0.49)^2 + (0.49)^2)$ and $\varepsilon = 10^{-11}$. The grid is refined according to the distance to the cylinder. The error distribution on the whole domain is presented in Fig. 11 while a zoom around the cylinder is presented in Fig. 12.

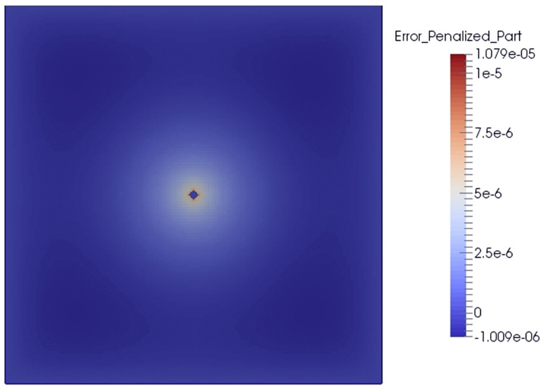

Fig. 11. Error distribution on the whole domain for the penalized cylinder with radius equal to 0.01.

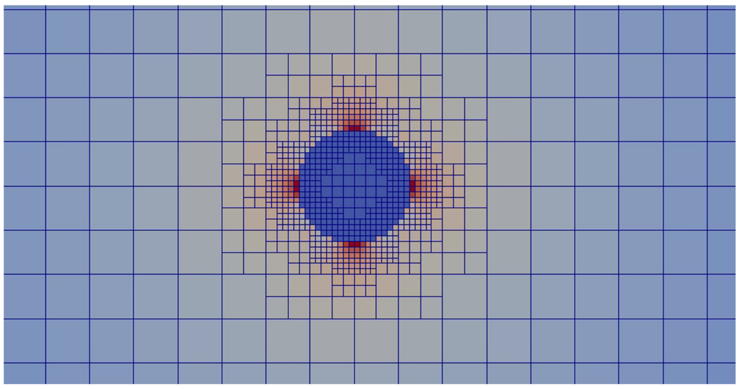

Fig. 12. Zoom on the error distribution around the penalized cylinder with radius equal to 0.01.

In what follows we compare the results obtained with our quadtree approach (AMR) to the results obtained with Cartesian uniform grids. The Cartesian results are obtained with the standard five-point stencil for the Laplacian operator with two different orderings: the Z-order (UZ) and the classical (i, j) order (US). The Laplace operator for the US ordering has a penta-diagonal structure while a more disperse structure is obtained for the UZ ordering. The non-uniform grid is obtained by mesh refinement near the cylinder with four levels of jumps between the maximal and the minimal depth of tree, enforcing balancing constraints through faces. In each case, the uniform grid is chosen to obtain the same degree of error than that obtained with the corresponding quadtree grid (see Table 5). Since maximal error is located near the cylinder boundary (first order penalization, see Fig. 12), the cell size for the uniform mesh is equal to the smaller mesh cell for the corresponding AMR quadtree grid. A comparison of the total number of points used for the uniform and quadtree grids is presented in Table 6. At level 15, the AMR grid has approximatively 220 times less grid points. The uniform grid is chosen so that infinite norm of the error is almost equal to that obtained with the corresponding quadtree grid. An intersecting consequence is that the L^2 norms are also equivalent whatever the mesh used. It is thus not necessary to use a fine mesh in the whole domain. The slight differences between US and UZ errors are a consequence of the ordering that has an influence on the linear solvers used. The CPU time required to solve the linear system is reported in Table 7 (see Fig. 13). All the tests presented in this section have been solved using 96 cores on 4 nodes. The computational time needed for the AMR are one to two orders of magnitude smaller compared to the US grid. The computational time to solve the linear system for the same uniform grid can be significantly larger for the UZ ordering than that for the US Cartesian ordering. The Krylov space used for this test is BCGS with ASM preconditioning and ILU sub-preconditioner.

The results of this section strongly depend on the configuration studied, *i.e.* the ratio between the square size and the cylinder diameter. On the one hand, even though Z-order may significantly reduce performance of linear solvers, the number of grid points can be reduced to an extent that makes the solution by far faster. On the other hand, if one is interested in solving a linear system (at least a Poisson equation) for a non-multi-scale problem, the Cartesian grid with the usual Cartesian ordering is the most efficient approach.

6.5. *Diffusion coefficient discontinuity*

We consider now the full problem given by equations (1). The diffusion coefficient $\kappa(x)$ is considered to be piecewise constant:

Table 5
Errors obtained with a quadtree structure (AMR with four jumps of level) and with uniform grids for two different orderings (natural Cartesian, US, and Z-ordering, UZ). The size of the uniform grids is chosen to obtain similar errors with respect to the quadtree case.

Tree level	L^2 US	L^∞ US	L^2 UZ	L^∞ UZ	L^2 AMR	L^∞ AMR
7	5.25e−06	3.49e−05	9.30e−06	5.72e−05	–	–
8	4.15e−06	2.79e−05	1.31e−06	1.29e−05	–	–
9	8.96e−07	9.20e−06	2.62e−06	1.81e−05	–	–
10	9.56e−07	9.09e−06	1.08e−06	1.08e−05	3.63e−06	1.07e−05
11	6.57e−07	5.75e−06	7.71e−07	8.01e−06	7.47e−07	7.99e−06
12	3.7e−07	4.16e−06	3.05e−07	2.87e−06	2.08e−07	2.87e−06
13	1.64e−07	2.05e−06	–	–	1.08e−07	1.49e−06
14	6.36e−08	9.34e−07	–	–	5.58e−08	7.33e−07
15	1.13e−08	3.26e−07	–	–	1.16e−08	3.81e−07

Table 6
Comparison of the number of points for the quadtree (AMR) and uniform (US/UZ) grids.

Tree level	US/UZ	AMR
7	16384	–
8	65536	–
9	262144	–
10	1048576	4900
11	4194304	19012
12	16777216	76036
13	67108864	302776
14	268435456	1214512
15	1073741824	4863616

Table 7
Comparison of the computational time (in seconds) required to solve the linear problem for the quadtree (AMR) and uniform grids (US and UZ).

Tree level	US	UZ	AMR
7	3.5329e−02	8.2994e−02	–
8	2.6406e−02	2.9260e−01	–
9	6.5642e−02	1.8123e+00	–
10	5.1011e−01	1.7149e+01	3.7959e−02
11	3.7639e+00	2.0778e+02	1.1091e−01
12	2.8374e+01	2.3868e+02	4.1040e−01
13	1.9036e+02	–	2.4283e+00
14	1.2410e+03	–	2.2296e+01
15	7.6472e+03	–	2.7102e+02

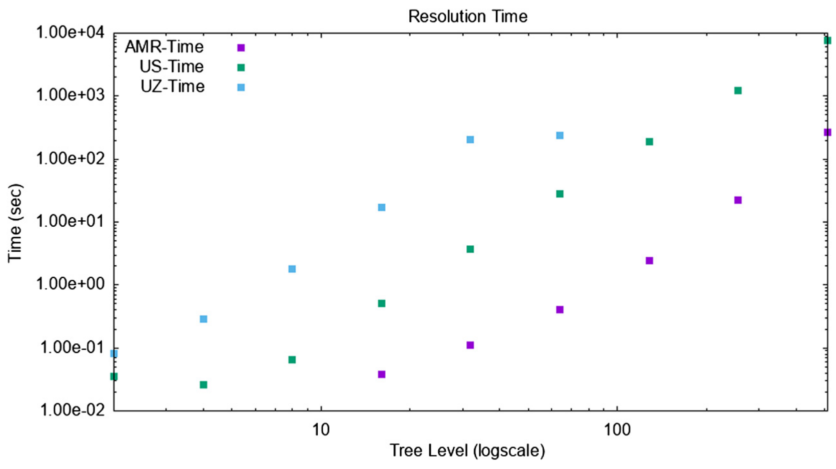

Fig. 13. Resolution time trend.

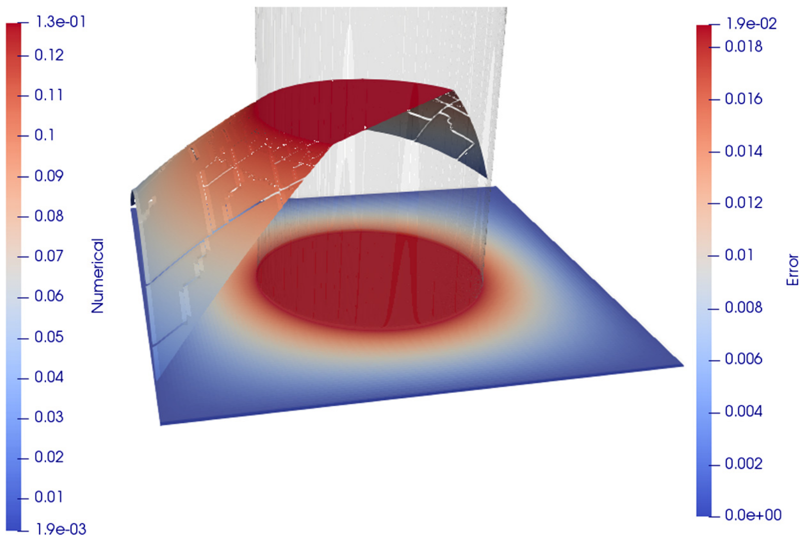

Fig. 14. Numerical result obtained by AMR, superposition of analytical and numerical solution plane section, with a projection of the error and a representation of the mollification (gray part). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

$$\kappa(x) = \begin{cases} 1 & \text{if } x \in G, \\ 100 & \text{if } x \in S. \end{cases}$$

We are thus interested in approximating the solution of the following system:

$$\kappa(x)\Delta u(x) = -1.0 \qquad \text{in } G \cup S, \tag{5a}$$

$$[\kappa(x)\partial_{\boldsymbol{n}} u(x)] = 0 \qquad \text{on } \gamma. \tag{5b}$$

The whole domain under consideration is the unit square $[0, 1] \times [0, 1]$. The interface γ separating subdomains G and S is defined by the circle with radius equal to $R = 0.25$ centered at $(0.5, 0.5)$. The subdomain S is inside the leading disk (Fig. 14). The analytical solution considered is defined by:

$$u(x, y) = \frac{1}{8} - \frac{1}{4\kappa_G}\left((x-0.5)^2 + (y-0.5)^2\right) \qquad \text{in } G, \tag{6}$$

$$u(x, y) = \frac{1}{8} - \frac{1}{4\kappa_S}\left((x-0.5)^2 + (y-0.5)^2\right) - \frac{R^2}{4}\left(1 - \frac{1}{\kappa_S}\right) \qquad \text{in } S. \tag{7}$$

The coefficient κ is regularized with equation (4) where parameters α, β and σ are chosen such that:

$$\alpha + \frac{1}{2}(\beta - \alpha) = 49.5 + 1, \; \frac{\beta - \alpha}{2} = 49.5,$$
$$\sigma = 100.$$

In what follows two classes of mesh refinements, called AMR1 and AMR2 are used.

6.5.0.1. AMR1 The first class of meshes used are refined near the mollification region around the interface γ with the following criteria:

- the maximal depth of the tree is fixed at value from $M = 7$ to $M = 13$;
- the squared domain is initially uniformly meshed with level $M - 4$;
- from $M - 4$ to M the mollified function (4) is evaluated on each octant. If the octant lies on the regularization zone[3] it splits in four children;
- balance constraints are applied on the jump zones.

Examples of this kind of meshes are presented in Fig. 15. The corresponding convergence results are presented in Table 8.

[3] The regularization zone is defined by $|\nabla\kappa(\boldsymbol{x})| > \varepsilon$, where ε is a small parameter, here 10^{-3}.

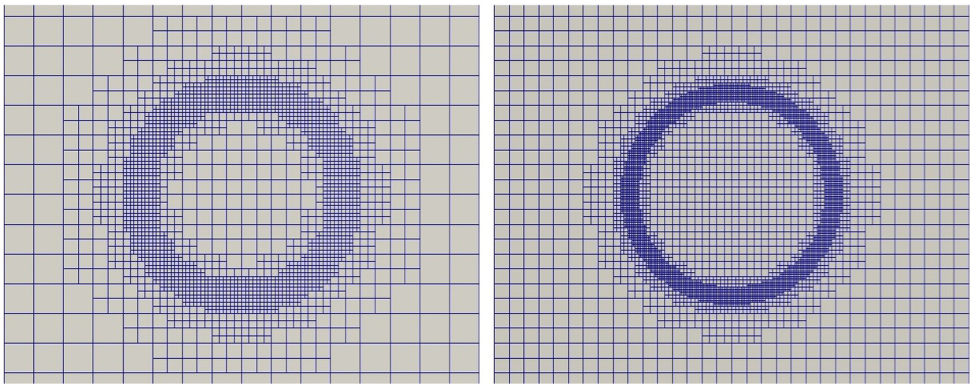

Fig. 15. Zoom on the AMR1 grids for levels 7 and 8.

Table 8
Convergence results for AMR1.

Tree level	Mesh points	L^∞	L^2
7	3784	2.40592e−02	1.69446e−01
8	14968	2.43815e−02	1.73977e−01
9	51472	3.86077e−02	2.77852e−01
10	112684	1.86597e−02	1.34336e−01
11	228484	2.87779e−03	1.98544e−02
12	465028	6.60242e−04	3.82096e−03
13	1124968	1.54341e−04	1.02991e−03

Table 9
Convergence results for AMR2.

Tree level	Mesh points	L^∞	L^2
7	16384	2.62959e−02	1.89502e−01
8	30868	2.56705e−02	1.84833e−01
9	59896	3.90077e−02	2.81516e−01
10	117796	1.88669e−02	1.36216e−01
11	233512	2.90886e−03	2.01399e−02
12	465028	6.60242e−04	3.82096e−03
13	1090300	1.53515e−04	1.02783e−03

6.5.0.2. AMR2 The second class of mesh refinement follows the criteria:

- the maximal depth of the tree is fixed at value from $M = 7$ to $M = 13$;
- the squared domain is initially uniformly meshed with $M = 7$ (16384 octants);
- from 7 to M the mollified function (4) is evaluated on each octant. If the octant lies on the regularization zone it splits in four children;
- balance constraints are applied on the jump zones.

Examples of this kind of meshes are presented in Fig. 16. The corresponding convergence results are presented in Table 9.

As expected, for both mesh refinements the overall error is distributed near the mollification region. The mesh can thus be relaxed outside the mollification region allowing to save CPU costs (time and memory). Compared to the balanced case, for given error level, we have here about 5 times less grid points. We study the same configuration described in the previous section but without the 2:1 balance constraint. An example of the unbalanced (4:1) grid is given in Fig. 17. Mesh convergence results are reported in Table 10.

A zoom of the unbalanced part is presented in Fig. 18.

The error distribution on the entire domain presented in Fig. 19 is comparable to the error obtained on balanced meshes for the same problem.

6.6. Three-dimensional problems

The extension of the numerical approach to 3D problems is quite straightforward. The number of the constraints to be satisfied for a consistent set of weights is 10 in 3D (6 in 2D). The numerical approach is still natively parallel.

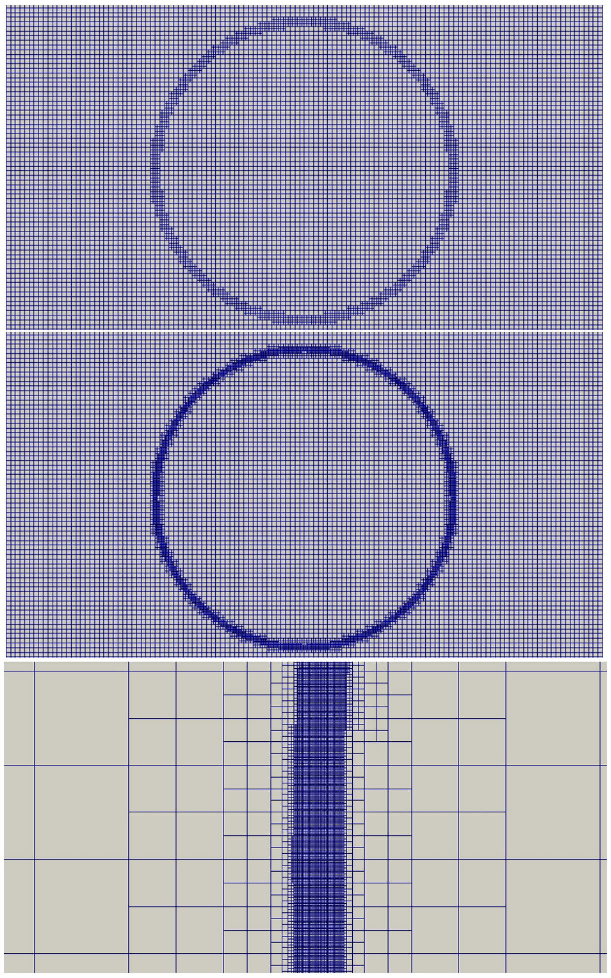

Fig. 16. Zoom on the AMR2 grids for levels 8 and 9 and 13.

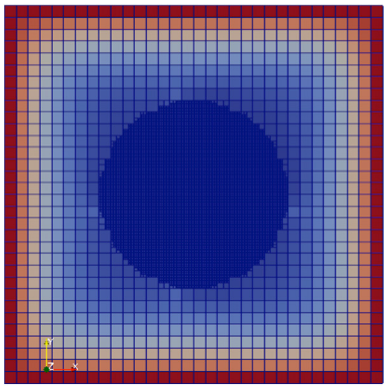

Fig. 17. Error example, level 8, sinus analytical function.

6.6.1. Validation

This section is devoted to study the convergence error of our approach for 3D problems. We consider a sinusoidal solution centered in the cube $[0, 1] \times [0, 1] \times [0, 1]$. One level of 2:1 mesh refinement is performed inside a fictitious sphere with radius equal to 0.15 centered in the computational domain.

Initially, the exact solution considered is $u_e(x, y, z) = \sin((x - 0.5)^2 + (y - 0.5)^2)$. It is invariant along the z-axis to check the 2D symmetry of the solution. Fig. 20 shows the matrix structure for 36128 grid points. The matrix presents several sub-blocks of non-zeros values coming from the Z-ordering. Note that the matrix structure is significantly different when

Table 10
Unbalanced test case. Sinus solution, first level 6.

Mesh points	L^∞	L^2	Order L^∞	Order L^2
796	0.00138827	0.00682539	–	–
3352	9.97279e−05	0.000550322	3.663	3.503
13588	2.73791e−05	0.000159641	1.846	1.764
54268	7.78934e−06	4.55914e−05	1.815	1.832
217936	1.95523e−06	1.14128e−05	2.032	2.027

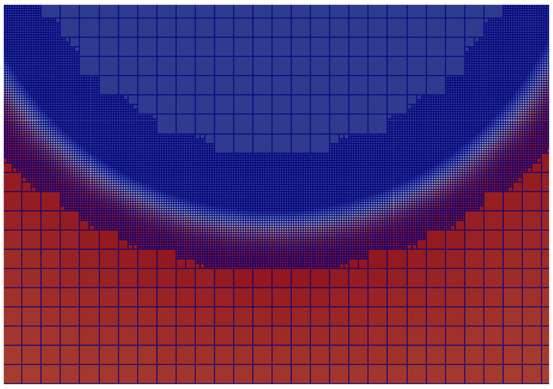

Fig. 18. Zoom on the unbalanced grid.

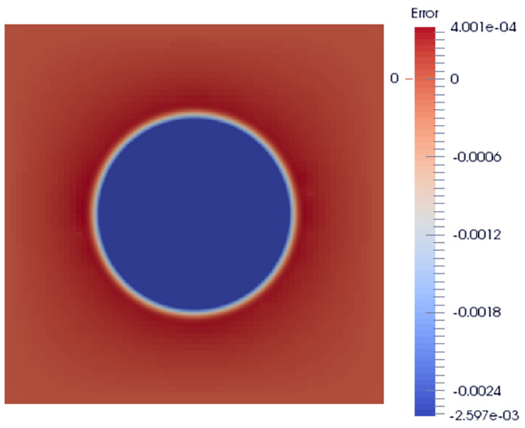

Fig. 19. Mesh referring to Table 11. Error distribution.

Table 11
Unbalanced AMR. First case level 7.

Mesh points	L^∞	L^2	Order L^∞	Order L^2
2464	1.91741e−02	4.09443e−02	–	–
10180	6.39336e−03	2.76724e−02	1.55	0.553
42928	2.59668e−03	1.45533e−02	1.252	0.899

a classical Cartesian ordering is used, even if no refinements are considered. Because of this, like in the 2D case, for the same number of grid points the computational time required to solve the linear system will be higher using the Z-ordering compared to the classical Cartesian ordering. Computational efficiency will be recovered since for given error, a significantly smaller resolution will be required.

An error map on a cross section is presented in Fig. 21. As expected, the error is concentrated in a narrow band around the fictitious sphere where refinement is performed.

Let Δ_h be the discretized laplacian operator and u_e the exact solution on mesh centers. The convergence order of the residual is computed as if the mesh were uniform: $p = 3 * \frac{\ln(err_1/err_2)}{\ln(np_2/np_1)}$, where *err* stands for error norm and *np* the total number of points. A convergence analysis of the residual $\Delta_h u_a - f$ is presented in Table 12. Second-order accuracy is

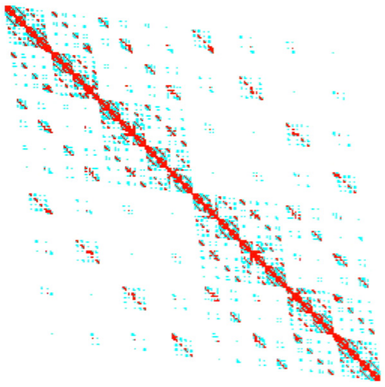

Fig. 20. Structure of the 3D matrix for the Laplacian operator with 2:1 graded grid (36128 points).

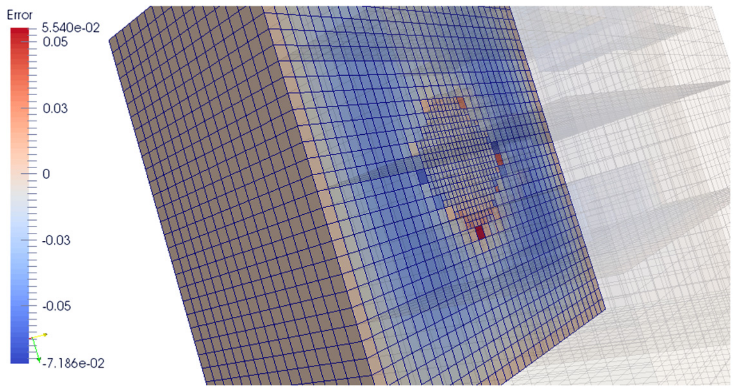

Fig. 21. Error map on a cross section for $u_e(x, y, z) = \sin((x - 0.5)^2 + (y - 0.5)^2)$ with 2:1 refinement in a sphere.

Table 12
Study of residual order.

Tree depth level	Mesh points	$\|\Delta_h u_a - f\|_\infty$	Order
5	4488	7.32456e−03	
6	36128	2.20676e−03	1.726
7	287680	5.98111e−04	1.888
8	2303400	1.5521e−04	1.945

obtained for the infinite norm. A map of the residuals in several cross sections is presented in Fig. 22. The smaller residuals are observed where the mesh is refined.

We now increase the level of jumps on a balanced mesh from one (previous example) to three. The error results are presented in Table 13. A second-order accuracy is almost reached for both the L^2 and L^∞ error norms. The distribution of the errors is presented in Fig. 23 for a mesh refinement with level 9 inside the sphere and 6 outside.

We now consider a 3D exact solution: $u_e(x, y, z) = \sin((x - 0.5)^2 + (y - 0.5)^2 + (z - 0.5)^2)$. The results in terms of error norms are reported in Table 14. The distributions of the error on the sphere and in a cross section without the sphere are presented in Fig. 24. A second order accuracy is obtained, and a symmetry on the error distribution is observed.

6.6.2. Random mesh generation

We applied a random algorithm to generate an arbitrary grid also for the three dimensional extension, see Fig. 25. The convergence order of the method is given in Table 15.

6.6.3. Penalization

We consider two test cases. The first one is the exact penalization, *i.e.* the exact solution is imposed on each node lying inside the centered sphere with radius equal to 0.05 by a penalty term. In the second one, the exact solution of the sphere boundary (that is constant for the solution under consideration) is reported on each nodes inside the sphere. The mesh is recursively refined in a narrow band around the sphere boundary.

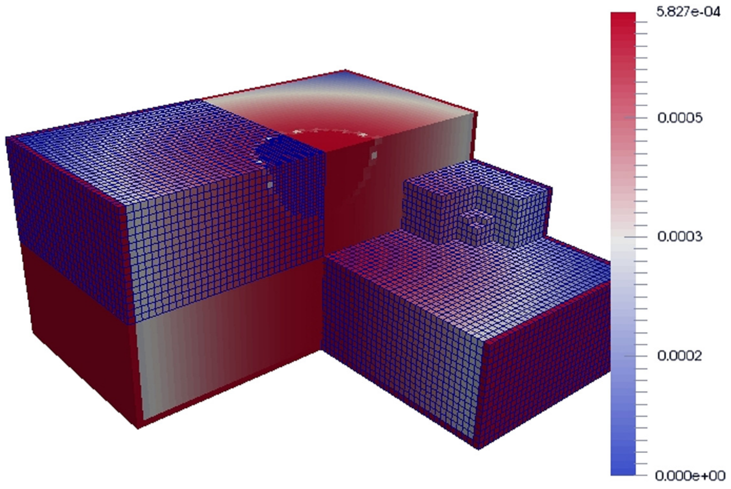

Fig. 22. Representation of the residuals for the 3D Laplace operator in several cross sections.

Table 13
Laplacian resolution with AMR in a sphere. Balanced mesh, three levels of difference between maximal and minimal depth, 2D sinus analytical function.

Mesh points	L^∞	L^2	Order L^∞	Order L^2
4880	2.41187e−04	1.16224e−04	–	–
32656	7.93562e−05	4.16123e−05	1.755	1.6184
264944	2.20059e−05	1.23492e−05	1.837	1.7461
2111880	5.82872e−06	3.36007e−06	1.919	1.8813
17103976	1.50078e−06	8.76136e−07	1.947	1.9281
137484026	3.8073e−07	2.23642e−07	1.976	1.965

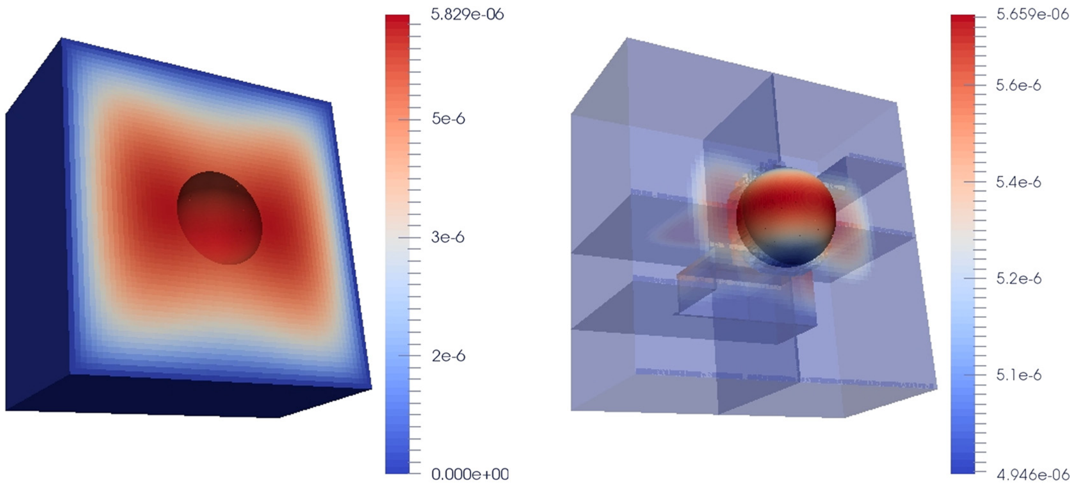

Fig. 23. Distribution of the errors for a mesh refinement with level 9 inside the sphere and 6 outside for 2D $u_e(x, y, z) = \sin((x-0.5)^2 + (y-0.5)^2)$.

Table 14
Laplacian resolution AMR in a sphere. Balanced mesh, three levels of difference between maximal and minimal depth. $u_e(x, y, z) = \sin((x-0.5)^2 + (y-0.5)^2 + (z-0.5)^2)$.

Mesh points	L^∞	L^2	Order L^∞	Order L^2
4880	3.9367e−04	1.97745e−04	–	–
32656	1.25677e−04	7.14724e−05	1.802	1.61
264944	3.54814e−05	2.12077e−05	1.812	1.74
2111880	9.42605e−06	5.77689e−06	1.916	1.878
17103976	2.42823e−06	1.50647e−06	1.945	1.935

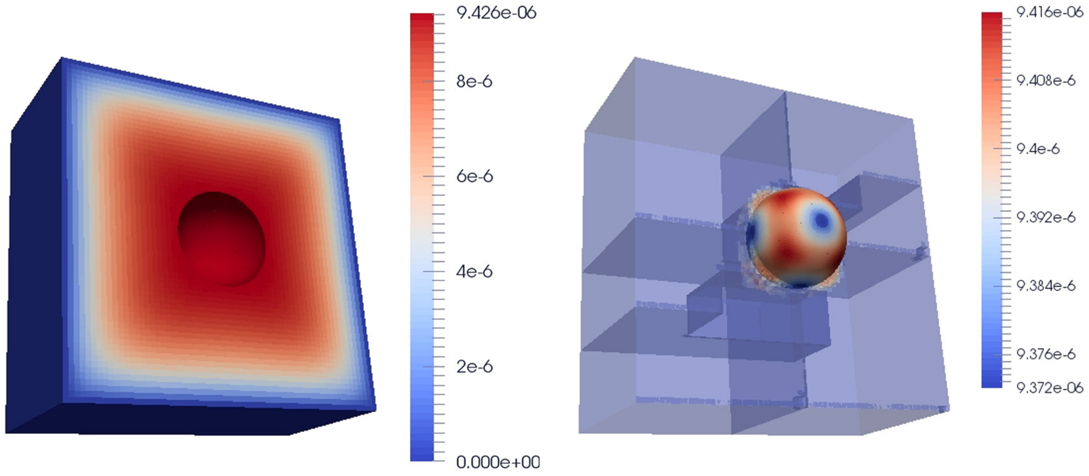

Fig. 24. Distribution of the errors for a mesh refinement with level 9 inside the sphere and 6 outside for 3D sinus analytical function.

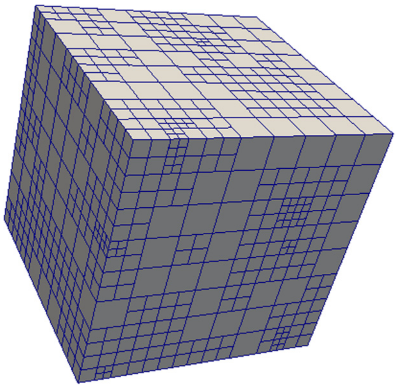

Fig. 25. Random mesh generation: initial grid.

Table 15
Convergence order for the random mesh generation. The first line involves 736 degrees of freedom, the last one 1573888.

Tree level	L^∞	L^2	Order L^∞	Order L^2
4	2.55285e−02	6.53591e−04	–	–
5	4.78024e−03	1.90869e−04	2.42	1.776
6	9.45935e−05	4.51667e−05	2.337	2.079
7	2.52537e−05	1.28204e−05	1.905	1.818
8	6.53981e−06	3.39493e−06	1.949	1.916

For the first test case, the error distribution in a cross section is presented in Fig. 26. The error is minimal near boundaries where the solution is explicitly imposed. Errors are presented in Table 16. As expected for an exact penalization on the sphere interior, a second order accuracy is reached.

For the second test case, corresponding to usual penalization, the error distribution in a cross section is presented in Fig. 27. The error is maximal near the internal boundary. Errors are presented in Table 17. The first order penalization is recovered.

7. Conclusions

A cell-centered finite-difference method to solve Poisson equation on hierarchical Cartesian meshes is proposed. The main idea is to minimize the local second-order truncation error coefficient by an appropriate choice of the discretization weights. In order to reduce communications, the stencil involves only the direct neighbors, *i.e.* all the cells that are in contact (with face, edge or corner) with the cell under consideration. The data structure is based on a linear octree and it is intrinsically parallel. Error analysis in two and three dimensions shows that this scheme is consistent with second-order accuracy. For multiscale problems, this approach outperforms uniform parallel Cartesian solvers since significantly less discretization

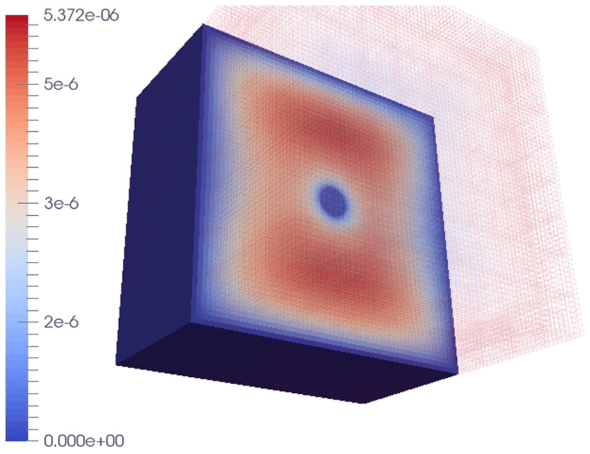

Fig. 26. Error distribution in a cross section for the 3D penalized sphere test case.

Table 16
Numerical errors and convergence for the second order penalized sphere.

Level	Mesh points	L^∞	L^2	Order L^∞	Order L^2
6	2976	0.000238244	0.000106971	–	–
7	10536	7.45466e−05	3.94184e−05	2.757	2.369
8	55672	2.43216e−05	1.19565e−05	2.019	2.15
9	380024	5.37193e−06	3.0825e−06	2.359	2.117
10	2826104	1.38687e−06	8.05033e−07	2.021	2.007
11	21907208	3.52385e−07	2.05731e−07	2.012	1.999

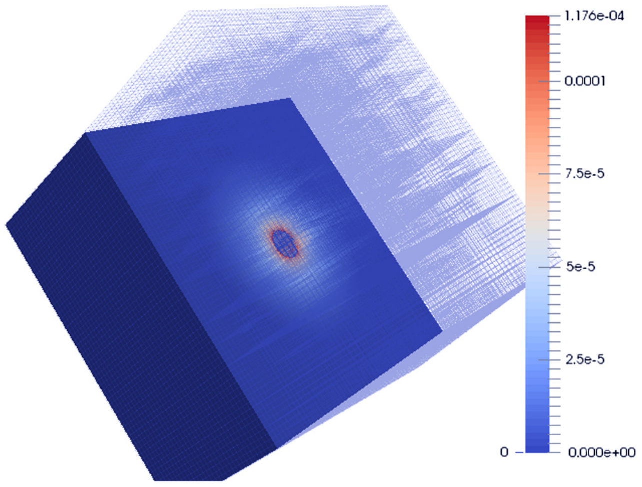

Fig. 27. Error distribution in a cross section for the 3D penalized sphere test case.

Table 17
Numerical errors and convergence for the first order penalized sphere.

Level	Mesh points	L^∞	L^2	Order L^∞	Order L^2
8	38256	1.30976e−03	1.50915e−05	–	–
9	330968	6.17589e−04	6.40046e−06	1.045	0.996
10	2646512	3.50834e−04	3.71911e−06	0.815	0.783
11	21205696	1.80231e−04	1.97858e−06	0.963	0.91

points are needed. Even if most of the examples presented in the paper deal with graded grids, it is shown that the scheme can be applied to non-graded grids without any modifications.

Acknowledgement

Experiments presented in this paper were carried out using the PLAFRIM experimental testbed, being developed under the Inria PlaFRIM development action with support from Bordeaux INP, LABRI and IMB and other entities: Conseil Régional d'Aquitaine, Université de Bordeaux and CNRS (and ANR in accordance to the programme d'investissements d'Avenir (see http://www.plafrim.fr/)).

References

[1] Philippe Angot, Charles-Henri Bruneau, Pierre Fabrie, A penalization method to take into account obstacles in incompressible viscous flows, Numer. Math. (ISSN 0945-3245) 81 (4) (1999) 497–520, https://doi.org/10.1007/s002110050401.
[2] Wolfgang Bangerth, Carsten Burstedde, Timo Heister, Martin Kronbichler, Algorithms and data structures for massively parallel generic adaptive finite element codes, ACM Trans. Math. Softw. 38 (2) (2011) 14.
[3] Carsten Burstedde, Lucas C. Wilcox, Omar Ghattas, p4est: scalable algorithms for parallel adaptive mesh refinement on forests of octrees, SIAM J. Sci. Comput. 33 (3) (2011) 1103–1133.
[4] Gilles Carbou, Pierre Fabrie, Boundary layer for a penalization method for viscous incompressible flow, Adv. Differ. Equ. 8 (12) (2003) 1453–1480.
[5] Frédéric Chantalat, Charles-Henri Bruneau, Cédric Galusinski, Angelo Iollo, Level-set, penalization and cartesian meshes: a paradigm for inverse problems and optimal design, J. Comput. Phys. 228 (17) (2009) 6291–6315.
[6] Sarah F. Frisken, Ronald N. Perry, Simple and efficient traversal methods for quadtrees and octrees, J. Graph. Tools 7 (3) (2002) 1–11.
[7] Arthur Guittet, Mathieu Lepilliez, Sebastien Tanguy, Frédéric Gibou, Solving elliptic problems with discontinuities on irregular domains – the Voronoi interface method, J. Comput. Phys. 298 (2015) 747–765.
[8] Louis H. Howell, John B. Bell, An adaptive mesh projection method for viscous incompressible flow, SIAM J. Sci. Comput. 18 (4) (1997) 996–1013.
[9] F. Jelassi, M. Azaïez, E. Palomo Del Barrio, A substructuring method for phase change modelling in hybrid media, Comput. Fluids 88 (Complete) (2013) 81–92, https://doi.org/10.1016/j.compfluid.2013.09.003.
[10] Hans Johansen, Phillip Colella, A Cartesian grid embedded boundary method for Poisson's equation on irregular domains, J. Comput. Phys. 147 (1) (1998) 60–85.
[11] Frank Losasso, Frédéric Gibou, Ron Fedkiw, Simulating water and smoke with an octree data structure, in: ACM Transactions on Graphics (TOG), vol. 23, ACM, 2004, pp. 457–462.
[12] Frank Losasso, Ronald Fedkiw, Stanley Osher, Spatially adaptive techniques for level set methods and incompressible flow, Comput. Fluids (ISSN 0045-7930) 35 (10) (2006) 995–1010, https://doi.org/10.1016/j.compfluid.2005.01.006, http://www.sciencedirect.com/science/article/pii/S004579300500174X.
[13] Chohong Min, Frédéric Gibou, Hector D. Ceniceros, A supra-convergent finite difference scheme for the variable coefficient Poisson equation on non-graded grids, J. Comput. Phys. 218 (1) (2006) 123–140.
[14] Mohammad Mirzadeh, Arthur Guittet, Carsten Burstedde, Frederic Gibou, Parallel level-set methods on adaptive tree-based grids, J. Comput. Phys. 322 (2016) 345–364.
[15] Guy M. Morton, A Computer Oriented Geodetic Data Base and a New Technique in File Sequencing, International Business Machines Company, New York, 1966.
[16] Maxim A. Olshanskii, Kirill M. Terekhov, Yuri V. Vassilevski, An octree-based solver for the incompressible Navier–Stokes equations with enhanced stability and low dissipation, Comput. Fluids 84 (2013) 231–246.
[17] Stéphane Popinet, Gerris: a tree-based adaptive solver for the incompressible Euler equations in complex geometries, J. Comput. Phys. 190 (2) (2003) 572–600.