Numer. Math. Theor. Meth. Appl.
doi: 10.4208/nmtma.2011.m12si02

Vol. **5**, No. 1, pp. 19-42
February 2012

Second Order Multigrid Methods for Elliptic Problems with Discontinuous Coefficients on an Arbitrary Interface, I: One Dimensional Problems

Armando Coco and Giovanni Russo*

Dipartimento di Matematica e Informatica, Università di Catania, Viale Andrea Doria, 6, 95125, Catania, Italy.

Received 16 December 2010; Accepted (in revised version) 05 May 2011

Available 21 December 2011

Abstract. In this paper we present a one dimensional second order accurate method to solve Elliptic equations with discontinuous coefficients on an arbitrary interface. Second order accuracy for the first derivative is obtained as well. The method is based on the Ghost Fluid Method, making use of ghost points on which the value is defined by suitable interface conditions. The multi-domain formulation is adopted, where the problem is split in two sub-problems and interface conditions will be enforced to close the problem. Interface conditions are relaxed together with the internal equations (following the approach proposed in [10] in the case of smooth coefficients), leading to an iterative method on all the set of grid values (inside points and ghost points). A multigrid approach with a suitable definition of the restriction operator is provided. The restriction of the defect is performed separately for both sub-problems, providing a convergence factor close to the one measured in the case of smooth coefficient and independent on the magnitude of the jump in the coefficient. Numerical tests will confirm the second order accuracy. Although the method is proposed in one dimension, the extension in higher dimension is currently underway [12] and it will be carried out by combining the discretization of [10] with the multigrid approach of [11] for Elliptic problems with non-eliminated boundary conditions in arbitrary domain.

AMS subject classifications: 35J25, 65N06, 65N55

Key words: Elliptic equation, discontinuous coefficient, second order accuracy, multigrid, arbitrary interface, jump conditions.

Introduction

Elliptic equations with jumping coefficients across a one-codimensional interface Γ arise in several applications. Let us mention as examples the steady-state diffusion problem in two materials with different diffusion coefficient separated by an arbitrary interface,

*Corresponding author. *Email address:* `russo@dmi.unict.it` (G. Russo) the Poisson equation coming from the projection method in incompressible Navier-Stokes equation for fluids with different density, the porous-media equation to model the oil reservoir, electrostatic problems, and many others. In order to close the problem, interface conditions related to the jump of the solution and of the flux across the interface are included. In all these problems the interface may be arbitrary (not aligned with a line grid) and can change in time.

Numerous techniques have been developed to treat such problem. Interface-fitted grid methods such as the ones based on Finite Elements Methods [3,5] are not suitable in case of moving interface, because a re-meshing grid is needed at each time step and this makes the computation expensive. Then an approach treating the interface embedded in a Cartesian grid and moving according to the velocity field of the fluid is preferred. Since the interface may not be aligned with the grid, a special treatment is needed. The simplest method makes use of the Shortley-Weller discretization [30], that discretizes the Laplacian operator with usual central difference away from the interface, and makes use of a non symmetric stencil in the points close to the interface, adding extra-grid points on Γ. While jumping condition on the solution is straightforward to discretize on interface points, the jump in the flux (involving the normal derivative) cannot be immediatly discretized in more than one dimension. In fact, Shortley-Weller discretization requires that the value of the normal derivative of the solution on both sides of the interface is suitably reconstructed at the intersection between the grid and the interface. This approach is adopted, for example, by Hackbusch in [20] to first order accuracy, and by other authors (see [6] and references therein) to second order accuracy. However, the method proposed by Bramble in [6] for second order accuracy is quite involved and not recommendable for practical purposes.

Methods based on embedding the domain in a Cartesian grid without adding extra-grid points are derived from the pioneering work of Peskin [26], where the Immersed Boundary Methods is introduced to model blood flows in the heart. In that paper a source term is localized on the the boundary and the method makes use of a discretized delta-function, leading to a first order accuracy. A second order accurate extension to jump coefficients is the Immersed Interface Methods, first developed by LeVeque and Li in [23]. Such method uses a six-point stencil to discretize the elliptic equation in grid points close to the interface Γ and the coefficients of such stencil are found by Taylor expansion of the solution. Jump conditions on the interface are then used to modify the coefficients appearing in the equation corresponding to nodes near Γ, in such a way that the overall discretization is second order accurate. Non-homogeneous jump conditions are allowed on the function and on the normal flux.

Another method which achieves second order accuracy by modifying standard difference formulas was proposed by Mayo in [24] for solving Poisson or biharmonic equation on irregular domains. Such method embeds the irregular domain in a regular region with a Cartesian grid and discretizes the equation on the whole region, by suitable extension of the solution outside.

In all these methods the only unknowns are the values on the grid points and the stencil may cross the interface, leading to a quite involved procedure to reach the desired accuracy, since the derivative of the solution may jump crossing the interface and values from the other side are used in the computation.

A rather simple method to use standard five-point stencil even close to the interface is the Ghost-Fluid Method, introduced by Fedkiw *et al.* in [14]. Here the authors point out that a two-phase problem could be reduced in two sub-problems by a multi-domain formulation, and each sub-problem may be discretized with the same technique used to solve a single problem with Dirichlet/Neumann boundary conditions. Such method makes use of extra grid points (*ghost points*) outside the domain in order to keep unchanged the symmetry of the stencil even for inside points close to the interface. In ghost points, interface conditions are enforced in order to close the discrete system.

Methods based on ghost points are discussed in [16], where Gibou *et al.* proposed a second-order accurate method for Dirichlet conditions on regular Cartesian grid. The value at the ghost nodes is assigned by linear extrapolation, and the whole discretization leads to a symmetric linear system, easily solved by a preconditioned conjugate gradient method. A fourth order accurate method is also proposed in [17]. Other methods use a non-regular Cartesian grid, such as in [9], where Gibou *et al.* present finite difference schemes for solving the variable coefficient Poisson equation and heat equation on irregular domains with Dirichlet boundary conditions, using adaptive Cartesian grids. One efficient discretization based on cut-cell method to solve more general Robin conditions is proposed by Gibou *et al.* in [25], which provides second order accuracy for the Poisson and heat equation and first order accuracy for Stefan type problems.

Other approaches based on cut-cell methods obtained by a Finite Volume discretization are presented in [22]. Cells that are cut by the boundary requires a special treatment, such as cell-merging and rotated-cell, in order to avoid a too strict restriction of the time step dictated by the CFL condition.

Several methods have been also proposed to model the interaction between multiphase flows and solid obstacles, such as Arbitrary Lagrangian Eulerian (ALE) [13, 15], Distribute Lagrangian Multiplier (DLM) [18], penalization methods [2, 29]. In [8] a combination of penalization and level-set methods is presented to solve inverse or shape optimization problems on uniform Cartesian meshes. In [32] Zhou *et al.* proposed a Matched Interface and Boundary (MIB) method for elliptic problems with sharp-edged interfaces.

In time-dependent problems requiring the solution of an elliptic problem at each time step an iterative solver is preferred with respect to a direct problem, since a good initial guess (the solution at the previous time step) is provided. Most iterative method for jumping coefficient are based on Domain Decomposition Methods [27], either with or without overlapping. Such methods are based on the multi-domain formulation, i.e., the problem is split in two sub-problems and interface conditions are enforced to achieve two sub-problems with respectively Dirichlet and Neumann boundary (coupled) conditions on the interface/boundary. Each sub-problem is solved and the solution at the interface is used to provide an updated right-hand side for the other sub-problem, and so on iteratively. A drawback of this method is that association between the Dirichlet/Neumann boundary condition and the sub-domain cannot be arbitrary (see [27, pag. 12]).

Most applications require second order accuracy in the gradient: for example, in projection method for incompressible Navier-Stokes equation, the gradient of the pressure is used to correct the fictitious velocity field leading it to satisfy the free-divergence condition. Also high-order accuracy [17] may be required, for instance when turbulence and shock interact, or high frequency wave propagation are presented in inhomogeneous media [4].

In [10] a second-order accurate discretization for elliptic problems in arbitrary domain and mixed boundary condition is provided, together with a convergence proof for the iterative solver for first order accuracy. The method is based on transforming the stationary problem into a fictitious evolutionary problem, both inside the domain and on the boundary. The problem is then discretized on a regular grid using non eliminated boundary conditions to determine the proper relaxation equation for the ghost points. The whole procedure is made efficient by a multigrid technique, as illustrated in [11].

The present paper provides a second order discretization of the problem based on the ghost-point method on regular Cartesian grid described in [10] and makes use of an iterative solver whose convergence is speeded up by a multigrid approach [11]. Interface conditions are neither eliminated from the discrete system (they are strongly coupled and their elimination is too hard to perform in more than one dimension) nor directly enforced (which leads to a non-convergent iterative method): they are relaxed together with the interior equations. This leads us to an iterative scheme for the set of all unknowns (internal points and ghost points). The method works also for non-homogeneous interface conditions. Although this paper provides a 1D description of the method, the generalization of the approach in higher dimension is currently underway [12] and can be obtained in an almost straightforward manner combining results from [10, 11].

Several multigrid approaches exist in literature to treat the jumping coefficient problem in 2D when the interface is aligned with the Cartesian grid. We mention the method based on operator-dependent interpolation [1, 21], where the interpolation is carried out by exploiting the continuity of the flux instead of the gradient of the solution, and the method based on Galerkin Coarse Grid Operator [28], which makes the algebraic problem more expensive from a computational point of view and does not take advantage from the fact that the discrete problem comes from a continuous problem.

In our approach we use the standard interpolation operator and discretize the operator in the coarser grid in the same way as in the fine grid, without making use of Galerkin conditions. But, since the defect may jump crossing the interface, a separated restriction for both sub-problems is needed, as performed in [11] for arbitrary domain with mixed boundary condition (without jumping coefficient). This approach provides a good convergence factor, comparable with ones measured for no-jumping case. We also show that the convergence factor does not depend on the magnitude of the jump in the coefficient. Interface conditions are relaxed, then have to be transferred to the coarse grid as well. In one-dimensional case this task is trivial, since such conditions are just two real values that can be copied to the coarse grid. In higher dimension interface conditions are stored in ghost points, which can show a complex structure for arbitrary interface. The restriction of interface condition defect can be carried out in the same manner of the restriction of boundary condition defect described in [11] for problems with non-eliminated boundary conditions: the defect is first extrapolated outside the domain and then transferred to the coarse grid in the same manner as the restriction of the defect of inside equations, i.e., without using values from the other side of the boundary. This work is currently underway.

The rest of the paper is divided in 3 sections. In the first section we describe the second order accurate discretization of the model problem and the iterative scheme obtained by the relaxation of the interface conditions. The second section is devoted to the multigrid approach, with a care description of the transfer operators. In Section 3 some numerical test is performed, to show the second order accuracy in the solution and in its first derivative as well. We measure also the convergence factor and compare it with the convergence factor obtained by other methods.

1. Second order accurate discretization

In this section we obtain a second order accurate numerical method to solve an elliptic equation with discontinuous coefficients. After introducing the model problem, we provide a discretization and an iterative solver of the linear system. In some applications one may be interested in second order accuracy also for the derivative of the solution. In numerical tests of Section 3 we show that the method is second order accurate in the solution and in its first derivative.

1.1. Model problem

Let us consider the model problem

$$-\frac{d}{dx}\left(\gamma\frac{du}{dx}\right)=f \quad \text{in } \Omega=[0,1], \tag{1.1a}$$

$$u(0)=g_0, \quad u(1)=g_1, \tag{1.1b}$$

where the diffusion coefficient $\gamma\colon [0,1]\to\mathbb{R}$ jumps on an interface $\alpha\in]0,1[$, i.e., is a smooth function in $[0,\alpha[$ and in $]\alpha,1]$, but may be discontinuous across α. We assume $\gamma>\epsilon>0$ in all the domain. If we solve this problem by standard central differences on a uniform grid, the accuracy of the method degrades to first order.

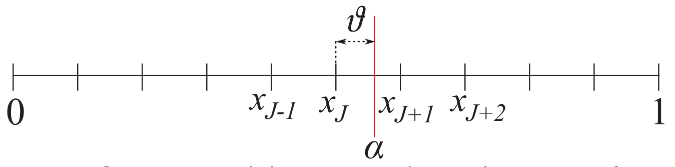


Figure 1: Computational domain Ω with an arbitrary interface α.

Let

$$u^L=u|_{[0,\alpha[}, \quad u^R=u|_{]\alpha,1]}, \quad \gamma^L=\gamma|_{[0,\alpha[}, \quad \gamma^R=\gamma|_{]\alpha,1]}$$

be the restriction functions of the solution and of the coefficient on the two subdomains.

We split the problem into the following subproblems:

$$-\frac{d}{dx}\left(\gamma^L \frac{du^L}{dx}\right) = f \quad \text{in } [0,\alpha[, \tag{1.2a}$$

$$u^L(0) = g_0, \tag{1.2b}$$

$$-\frac{d}{dx}\left(\gamma^R \frac{du^R}{dx}\right) = f \quad \text{in }]\alpha,1], \tag{1.3a}$$

$$u^R(1) = g_1. \tag{1.3b}$$

In order to close the problem, we must provide an additional boundary condition for each of u^L and u^R on the interface α. This additional conditions are inferred to the requirement that the solution u and the flux $\gamma u'$ are continuous across α. Introducing the *jumping operator* on α

$$[w] = \lim_{x\to\alpha^+} w - \lim_{x\to\alpha^-} w,$$

the additional boundary conditions may be resumed as

$$[u] = 0, \quad [\gamma u'] = 0$$

and are called *transmission conditions* [27]. They can be inferred by a physical requirement: for instance, in steady-state diffusion problems in two materials, the temperature and its flux are required to be continuous across α. Non-homogeneous interface conditions may appear, for example, in presence of a delta-function on the right hand side $f = f_1 + \delta_\alpha$, with $f_1 \in C^0([0,1])$. Precisely, the two following problems are equivalent:

$$\begin{aligned} -\frac{d}{dx}(\gamma \frac{du}{dx}) &= f_1 + C\delta_\alpha \quad \text{in } [0,1], \\ u(0) = g_0, &\quad u(1) = g_1, \end{aligned}$$

and

$$\begin{aligned} -\frac{d}{dx}(\gamma \frac{du}{dx}) &= f_1 \quad \text{in } [0,\alpha[\cup]\alpha,1], \\ u(0) = g_0, &\quad u(1) = g_1, \\ [u] = 0, &\quad [\gamma\, u'] = -C. \end{aligned}$$

In the following we suppose the right-hand side is a regular function in the two sub-regions, and non-homogeneous interface conditions are allowed:

$$[u] = g_D, \quad [\gamma u'] = g_N. \tag{1.4}$$

Such general case is relevant for some applications, for example pressure equation for incompressible flow in presence of surface tension at the interface.

The two subproblems (1.2) and (1.3) are then coupled on α and cannot be solved separately. The whole problem becomes

$$-\frac{d}{dx}\left(\gamma^L \frac{du^L}{dx}\right) = f \quad \text{in } [0,\alpha[\ , \tag{1.5}$$

$$-\frac{d}{dx}\left(\gamma^R \frac{du^R}{dx}\right) = f \quad \text{in }]\alpha,1]\ , \tag{1.6}$$

$$u^L(0) = g_0, \qquad u^R(1) = g_1, \tag{1.7}$$

$$[u] = g_D, \qquad [\gamma u'] = g_N. \tag{1.8}$$

1.2. Discretization

Let N be an integer, $h = 1/(N+1)$ be the spatial step and $x_0, x_1, \cdots, x_N, x_{N+1}$ be the equally spaced grid points, with $x_j = jh$. Let J be such that $x_J \le \alpha < x_{J+1}$ (see Fig. 1). We write $J = \lfloor \alpha \rfloor$, where $\lfloor \cdot \rfloor$ denotes the integer part. We will denote by $\mathscr{L}_j[w]$ the quadratic interpolant of w in nodes $\left\{x_{j-1}, x_j, x_{j+1}\right\}$. By u_j^L $[u_j^R]$ we denote the component of the numerical solution which approximates $u^L(x_j)$ $[u^R(x_j)]$, while we intend

$$f_j = f(x_j), \quad \gamma_j^L = \gamma^L(x_j), \quad \gamma_j^R = \gamma^R(x_j).$$

Let us discretize the system (1.7). Discretizing Eq. (1.5) on nodes $x_1, x_2, \cdots, x_J$ using central differences for the solution u^L and linear interpolation for the coefficient function γ^L, we obtain:

$$\frac{1}{h^2}\left(\gamma^L_{j-\frac{1}{2}}\left(u_j^L - u_{j-1}^L\right) + \gamma^L_{j+\frac{1}{2}}\left(u_j^L - u_{j+1}^L\right)\right) = f_j, \ \ j = 1, \cdots J, \tag{1.9}$$

where

$$\gamma^L_{j+1/2} = \frac{1}{2}(\gamma_j^L + \gamma_{j+1}^L).$$

In Eq. (1.9) for $j = 1$ the value u_0^L is given by the Dirichlet condition (1.7): $u_0^L = g_0$. It can be easily eliminated from (1.9), but we will leave it in the system just for simplicity. The same applies for u_N^R discretizing Eq. (1.6) in node x_N.

Eq. (1.9) for $j = J$ needs to know the values of u^L and γ^L in node x_{J+1}. Since u' and γ are discontinuous, we cannot use respectively u_{J+1}^R and γ_{J+1}^R, because this may result in a loss of accuracy, since it smears out the coefficient γ and the numerical solution itself, while both jump on the interface. Then we need to add an additional grid point value for the numerical solution $u^L(x_{J+1})$, called *ghost point value*, and to extrapolate γ^L up to the first ghost point x_{J+1}. The same argument holds for u^R and γ^R in their ghost point x_J, when discretizing Eq. (1.6) in node x_{J+1}.

The unknowns of the numerical method are therefore the $N+4$ quantities

$$u_0^L, \cdots, u_{J+1}^L, u_J^R, \cdots, u_{N+1}^R. \tag{1.10}$$

This approach has been called *Ghost Fluid Method* and used in the context of multi-fluid flows [14]. The two additional unknowns u^L_{J+1} and u^R_J require two additional boundary conditions to close the system, which are given by the transmission conditions (1.4), resulting in a 2×2 sub-system. We will not solve this sub-system for u^L_{J+1} and u^R_J, but we instead leave it in the whole linear system, which will be solved iteratively. The extrapolation for the coefficient functions γ^L and γ^R is simple linear extrapolation:

$$\gamma^L_{J+1} = 2\,\gamma^L_J - \gamma^L_{J-1}, \quad \gamma^R_J = 2\,\gamma^L_{J+1} - \gamma^L_{J+2}.$$

Using then central differences to discretize (1.5) and (1.6), linear and quadratic interpolation to discretize respectively the two conditions (1.8), we obtain the following second order $(N+4)\times(N+4)$ linear system:

$$u^L_0 = g_0, \tag{1.11}$$

$$\frac{1}{h^2}\left(\gamma^L_{j-\frac{1}{2}}\left(u^L_j - u^L_{j-1}\right) + \gamma^L_{j+\frac{1}{2}}\left(u^L_j - u^L_{j+1}\right)\right) = f_j, \qquad j = 1,\cdots,J, \tag{1.12}$$

$$\left((1-\vartheta)u^R_J + \vartheta u^R_{J+1}\right) - \left((1-\vartheta)u^L_J + \vartheta u^L_{J+1}\right) = g_D, \tag{1.13}$$

$$\gamma^R_\alpha\,\mathscr{L}'_J[u^R](\alpha) - \gamma^L_\alpha\,\mathscr{L}'_{J-1}[u^L](\alpha) = g_N, \tag{1.14}$$

$$\frac{1}{h^2}\left(\gamma^R_{j-\frac{1}{2}}\left(u^R_j - u^R_{j-1}\right) + \gamma^R_{j+\frac{1}{2}}\left(u^R_j - u^R_{j+1}\right)\right) = f_j, \qquad j = J+1,\cdots,N, \tag{1.15}$$

$$u^R_N + 1 = g_1, \tag{1.16}$$

with γ^L_α and γ^R_α obtained by linear interpolation:

$$\gamma^L_\alpha = (1-\vartheta)\gamma^L_J + \vartheta\gamma^L_{J+1}, \quad \gamma^R_\alpha = (1-\vartheta)\gamma^R_J + \vartheta\gamma^R_{J+1}$$

and $\vartheta = (\alpha - x_J)/h \in [0,1]$.

If we apply a simple iterative method such as Gauss-Seidel or Jacobi to this linear system, in general it will not converge, unless we solve the 2×2 sub-system of transmission conditions, eliminating them from the whole system. This elimination is easy to perform in one dimension, but becomes quite involved in higher dimension. Therefore, we prefer to work with the whole linear system without eliminate transmission conditions from it, in order to extend the method to higher dimension in a forthcoming paper [12]. Then we have to find a different approach to solve iteratively the previous linear system. This can be done by relaxing the transmission conditions.

1.3. Iterative method

In order to find a convergent iterative method to solve the linear system (1.11)-(1.16), following the approach introduced in [10] we solve the *associate time-dependent* problem in the unknowns $u^L(x,t)$ and $u^R(x,t)$ for $(x,t)\in[0,1]\times(0,+\infty)$:

$$u^L(0,t)=g_0, \tag{1.17}$$

$$\frac{\partial u^L}{\partial t}=\mu\left(\frac{\partial}{\partial x}\left(\gamma^L\frac{\partial u^L}{\partial x}\right)+f\right),\quad x\in[0,\alpha[\,, \tag{1.18}$$

$$\left.\frac{\partial u^L}{\partial t}\right|_{x=\alpha}=\mu_N\left(\left[\gamma\frac{\partial u}{\partial x}\right]-g_N\right), \tag{1.19}$$

$$\left.\frac{\partial u^R}{\partial t}\right|_{x=\alpha}=\mu_D\left(g_D-[u]\right), \tag{1.20}$$

$$\frac{\partial u^R}{\partial t}=\mu\frac{\partial}{\partial x}\left(\gamma^R\frac{\partial u^R}{\partial x}\right)+f,\quad x\in]\alpha,1]\,, \tag{1.21}$$

$$u^R(1,t)=g_1, \tag{1.22}$$

where μ is a positive function, and μ_D and μ_N are two positive constants, that will be set in Section 1.4 to satisfy some stability condition.

The choice of the sign of the two constants μ_D and μ_N is crucial and requires some explanation. Roughly speaking, when replacing a vector equation $F(w)=0$ for $F:\mathbb{R}^m\to\mathbb{R}^m$ by $\dot{\omega}=F(\omega)$, we have to be sure that the solution is asymptotically stable, i.e. that $\lambda(\nabla_\omega F)<0$. Eq. (1.20) will be used to compute u_J^R, therefore the derivative of the right hand side of Eq. (1.20) with respect to u_J^R has to be negative, to ensure convergence to equilibrium. Eq. (1.19) is used to determine u_{J+1}^L by a transport equation on $u^L(x,t)$. Since $x_{J+1}>\alpha$ the propagation speed $\mu_N\gamma^L$ associated to $u^L(x,t)$, has to be positive.

We are obviously interested in the steady-state solution and the time t represents an iterative parameter. We observe that transmission conditions (1.19) and (1.20) can be replaced by

$$\left.\frac{\partial u^R}{\partial t}\right|_{x=\alpha}=\mu_N\left(g_N-\left[\gamma\frac{\partial u}{\partial x}\right]\right),$$

$$\left.\frac{\partial u^L}{\partial t}\right|_{x=\alpha}=\mu_D\left([u]-g_D\right),$$

because both choices lead to the same steady state conditions.

To obtain a second order accurate solution in space we are allowed to discretize first order accurate the time derivative. Using forward Euler in time and central differences in space for (1.18) and (1.21), we obtain (superscripts L and R are omitted):

$$u_j^{(m+1)}=u_j^{(m)}+\mu_j\,\Delta t\left(f_j-\frac{\gamma_{j-\frac{1}{2}}\left(u_j^{(m)}-u_{j-1}^{(m)}\right)+\gamma_{j+\frac{1}{2}}\left(u_j^{(m)}-u_{j+1}^{(m)}\right)}{h^2}\right), \tag{1.23}$$

where $j = 1,\cdots,J$ for u^L and $j = J+1,\dots,N$ for u^R. Choosing the maximum time step allowed by the CFL condition for diffusion equation, i.e., $\mu_j\,\Delta t = h^2/(\gamma_{j+1/2}+\gamma_{j-1/2})$, Eq. (1.23) becomes:

$$u_j^{(m+1)} = \frac{1}{\gamma_{j-\frac{1}{2}}+\gamma_{j+\frac{1}{2}}}\left(f_j\,h^2+\gamma_{j-\frac{1}{2}}\,u_{j-1}^{(m)}+\gamma_{j+\frac{1}{2}}\,u_{j+1}^{(m)}\right), \tag{1.24}$$

where $j = 1,\cdots,J$ for u^L and $j = J+1,\cdots,N$ for u^R. Observe that such equation is the one obtained by applying Jacobi iteration to Eqs. (1.12) and (1.15).

Let us discretize Eq. (1.19). The time derivative is discretized by forward Euler at the ghost point x_{J+1}, which is the quantity we want to compute. The jump is discretized as in (1.14), so it is second order accurate. We obtain the iteration:

$$u_{J+1}^{L,(m+1)} = u_{J+1}^{L,(m)}+\mu_N\Delta t\left(\gamma_\alpha^R\,\mathscr{L}_J'[u^{R,(m)}](\alpha)-\gamma_\alpha^L\,\mathscr{L}_{J-1}'[u^{L,(m)}](\alpha)-g_N\right). \tag{1.25}$$

Likewise, in Eq. (1.20) we discretize the time derivative in x_J, obtaining:

$$\begin{aligned} u_J^{R,(m+1)} = u_J^{R,(m)}+\mu_D\,\Delta t\,\Big((1-\vartheta)u_J^{L,(m)}+\vartheta u_{J+1})^{L,(m)} \\ -(1-\vartheta)u_J^{R,(m)}+\vartheta u_{J+1}^{R,(m)}+g_D\Big). \end{aligned} \tag{1.26}$$

Iterations (1.24), (1.25) and (1.26) constitute the iterative scheme to solve problem (1.1) to second order accuracy.

1.4. Choosing constants μ_D and μ_N for transmission conditions

In (1.25) and (1.26) two arbitrary constants μ_D and μ_N appear. Following the same argument as in [10], such constants will be chosen in order to satisfy some stability condition for the equation where they appear. This procedure is not rigorous because it does not take into account the coupling between the equations, and does not consist in a convergence proof. However, in all numerical tests we performed, the conditions we find seem to guarantee convergence.

Constant μ_D is introduced in Eq. (1.20), which is just a relaxation of the jump condition. Then we require:

$$\mu_D\,\Delta t < 1. \tag{1.27}$$

This condition will ensure positivity, and is a factor 2 more stringent than just stability restriction. For practical purpose, we set $\mu_D\,\Delta t = 0.9$. In order to obtain a condition on μ_N, we rewrite Eq. (1.19) as follows (we have supposed for simplicity homogeneous jump $g_N = 0$):

$$\frac{\partial u^L}{\partial t}+\mu_N\,\gamma^L\,\frac{\partial u^L}{\partial x} = \mu_N\,\gamma^R\,\frac{\partial u^R}{\partial x}, \quad t\in(0,\infty). \tag{1.28}$$

This is a simple convection equation with speed $\mu_N\,\gamma^L$. Then a simple CFL condition for convection equation might be

$$\mu_N\Delta t \le \frac{h}{\gamma^L}.$$

Numerical experiments show that this condition is not enough, especially in the case $\gamma^R/\gamma^L \gg 1$. An explanation of this behavior may be that the right-hand side of (1.28) is not stationary when the convection evolves in time, but it depends on time itself by u^R. An acceptable condition is

$$\mu_N \Delta t \leq \frac{h}{\max\{\gamma^L,\gamma^R\}}. \tag{1.29}$$

For practical purpose we choose $\mu_N \Delta t = 0.9\, h/\max\left\{\gamma^L,\gamma^R\right\}$. Numerical tests show that conditions (1.27) and (1.29) are sufficient for guarantee convergence, but not necessary. A more detailed analysis is in progress.

Notice that $\mu \Delta t = \mathcal{O}(h^2)$, $\mu_N \Delta t = \mathcal{O}(h)$, $\mu_D \Delta t = \mathcal{O}(1)$. Furthermore, only the product of the constants times Δt enters into the conditions, therefore we may imagine that $\Delta t = 1$.

2. Multigrid approach

The convergence of the iterative method proposed in Sec. 1.3 is usually very slow. To accelerate the convergence we use a multigrid strategy. To make the iteration scheme (1.24)-(1.26) a building block for an efficient multi-grid solver, we must be sure that such iteration (*relaxation scheme*) has the *smoothing property*, i.e. that after few steps, the error becomes smooth (not necessarily small). Roughly speaking, the high-frequency components of the error reduce quickly. We do not explain all multigrid features, but just what is different from classical multigrid approach, remanding to the literature for more details (e.g., see [7,19,31]). The iteration scheme (1.24)-(1.26) is a Jacobi-like scheme, as mentioned in Sec. 1.3. Jacobi scheme is not a good smoother, since high-frequency components of the error reduce slowly. A good smoother is instead the Gauss-Seidel scheme. Then, we use a Gauss-Seidel version of (1.24)-(1.26) as relaxation scheme, i.e.

$$u_j^{L,(m+1)} = \frac{1}{\gamma_{j-\frac{1}{2}}+\gamma_{j+\frac{1}{2}}}\left(f_j\, h^2 + \gamma_{j-\frac{1}{2}}\, u_{j-1}^{L,(m+1)} + \gamma_{j+\frac{1}{2}}\, u_{j+1}^{L,(m)}\right), \quad j=1,\cdots,J, \tag{2.1}$$

$$u_{J+1}^{L,(m+1)} = u_{J+1}^{L,(m)} + \mu_N \Delta t\left(\gamma_\alpha^R\, \mathscr{L}_J'[u^{R,(m)}](\alpha) - \gamma_\alpha^L\, \mathscr{L}_{J-1}'[\tilde{u}^L](\alpha) - g_N\right), \tag{2.2}$$

$$\begin{aligned} u_J^{R,(m+1)} = u_J^{R,(m)} + \mu_D \Delta t\Big((1-\vartheta)u_J^{L,(m+1)} + \vartheta u_{J+1}^{L,(m+1)} \\ -(1-\vartheta)u_J^{R,(m)} - \vartheta u_{J+1}^{R,(m)} + g_D\Big), \end{aligned} \tag{2.3}$$

$$u_j^{R,(m+1)} = \frac{1}{\gamma_{j-\frac{1}{2}}+\gamma_{j+\frac{1}{2}}}\left(f_j\, h^2 + \gamma_{j-\frac{1}{2}}\, u_{j-1}^{R,(m+1)} + \gamma_{j+\frac{1}{2}}\, u_{j+1}^{R,(m)}\right), \quad j=J+1,\cdots,N, \tag{2.4}$$

where in (2.2) we intend $\tilde{u}^L$ such that $\tilde{u}_j^L = u_j^{L,(m+1)}$ for $j < J+1$ and $\tilde{u}_{J+1}^L = u_{J+1}^{L,(m)}$. The unknowns are updated in the same order reported in (1.10).

In order to explain the multigrid approach, we just describe the two-grid correction scheme (TGCS), because all the other schemes, such as V-cycle, W-cycle, F-cycle or Full

Multigrid cycle, can be easily derived from it (see [31, Sections 2.4 and 2.6] for more details). Let us introduce some notation. For a grid of spatial step h, we denote:

$$
\begin{aligned}
&J = \left\lfloor \frac{\alpha}{h} \right\rfloor, \quad \vartheta = \frac{\alpha}{h} - J,\\
&S(\Omega_h) = \left\{ \mathbf{w}_h = (w^L, w^R) \text{ such that } w^L\colon \{x_0, \dots, x_{J+1}\} \to \mathbb{R},\ w^R\colon \{x_J, \dots, x_{N+1}\} \to \mathbb{R} \right\},\\
&\overset{\circ}{S}(\Omega_h) = \left\{ \mathbf{w}_h = (w^L, w^R) \text{ such that } w^L\colon \{x_1, \dots, x_J\} \to \mathbb{R},\ w^R\colon \{x_{J+1}, \dots, x_N\} \to \mathbb{R} \right\},\\
&\mathbf{u}_h = ((u^L_j)_{j=0,\dots,J+1}, (u^R_j)_{j=J,\dots,N+1}) \in S(\Omega_h),\\
&\gamma_h = ((\gamma^L_j)_{j=0,\dots,J+1}, (\gamma^R_j)_{j=J,\dots,N+1}) \in S(\Omega_h),\\
&\mathbf{f}_h \in \overset{\circ}{S}(\Omega_h) \text{ such that } \mathbf{f}_h(x_j) = f_j,\\
&L_h\colon S(\Omega_h) \times S(\Omega_h) \longrightarrow \overset{\circ}{S}(\Omega_h) \text{ such that}\\
&\left(L_h(\gamma_h, \mathbf{u}_h)\right)_j = \frac{1}{h^2} \left(\gamma^L_{j-\frac{1}{2}} \left(u^L_j - u^L_{j-1} \right) + \gamma^L_{j+\frac{1}{2}} \left(u^L_j - u^L_{j+1} \right) \right) \text{ if } j \le J,\\
&\left(L_h(\gamma_h, \mathbf{u}_h)\right)_j = \frac{1}{h^2} \left(\gamma^R_{j-\frac{1}{2}} \left(u^R_j - u^R_{j-1} \right) + \gamma^R_{j+\frac{1}{2}} \left(u^R_j - u^R_{j+1} \right) \right) \text{ if } j \ge J+1,\\
&[\,\cdot\,]^D_h : S(\Omega_h) \longrightarrow \mathbb{R} \text{ such that}\\
&\left[\mathbf{u}_h\right]^D_h = \left((1-\vartheta) u^R_J + \vartheta u^R_{J+1} \right) - \left((1-\vartheta) u^L_J + \vartheta u^L_{J+1} \right),\\
&[\,\cdot\,,\cdot\,]^N_h : S(\Omega_h) \times S(\Omega_h) \longrightarrow \mathbb{R} \text{ such that}\\
&\left[\gamma_h, \mathbf{u}_h\right]^N_h = \gamma^R_\alpha \mathscr{L}'_J[u^R](\alpha) - \gamma^L_\alpha \mathscr{L}'_{J-1}[u^L](\alpha).
\end{aligned}
$$

The linear system (1.11)-(1.16) can be resumed as follows:

$$
\begin{aligned}
&L_h(\gamma_h, \mathbf{u}_h) = \mathbf{f}_h, && (2.5)\\
&\left[\mathbf{u}_h\right]^D_h = g_D, && (2.6)\\
&\left[\gamma_h, \mathbf{u}_h\right]^N_h = g_N, && (2.7)\\
&u^L_0 = g_0, && (2.8)\\
&u^R_N = g_1. && (2.9)
\end{aligned}
$$

For simplicity we assume that $N+1 = 1/h$ is a power of 2. The TGCS consists into the following algorithm:

Algorithm 2.1. *1. Set initial guess* $\mathbf{u}_h = 0$.

2. Relax ν_1 *times on the finest grid: for k from* 1 *to* ν_1 *do* (2.1), (2.2), (2.3).

3 *Compute the defects* $\mathbf{r}_h \in \overset{\circ}{S}(\Omega_h)$, $\tilde{g}_D, \tilde{g}_N \in \mathbb{R}$:

$$\begin{aligned}
&\mathbf{r}_h = \mathbf{f}_h + L_h(\gamma_h, \mathbf{u}_h), \\
&\tilde{g}_D = g_D - \left[\mathbf{u}_h\right]_h^D, \\
&\tilde{g}_N = g_N - \left[\gamma_h, \mathbf{u}_h\right]_h^N.
\end{aligned}$$

4 *Transfer the defect* $\mathbf{r}_h$ *to a coarser grid with spatial step* $2h$ *by a suitable restriction operator*

$$\mathbf{r}_{2h} = I_{2h}^{h}\left(\mathbf{r}_h\right).$$

5 *Solve exactly the residual problem on the coarser grid in the unknow* $\mathbf{e}_{2h} \in S(\Omega_{2h})$

$$\begin{aligned}
&L_h(\gamma_{2h}, \mathbf{e}_{2h}) = \mathbf{r}_{2h}, \\
&\left[\mathbf{e}_{2h}\right]_h^D = \tilde{g}_D, \\
&\left[\gamma_{2h}, \mathbf{u}_{2h}\right]_h^N = \tilde{g}_N, \\
&e_0^L = 0, \\
&e_{(N+1)/2}^R = 0.
\end{aligned}$$

6 *Transfer the error to the finest grid by a suitable interpolation operator*

$$\mathbf{e}_h = I_h^{2h}\left(\mathbf{e}_{2h}\right).$$

7 *Correct the fine-grid approximation*

$$\mathbf{u}_h = \mathbf{u}_h + \mathbf{e}_h.$$

8 *Relax* ν_2 *times on the finest grid: for* k *from* 1 *to* ν_2 *do* (2.1)-(2.3).

To complete the description of TGCS, we have just to explain the steps concerning grid migration (steps 4 and 6).

2.1. Transfer grid operators

In this section, we describe the transfer grid operators for vertex-centered grid. Observe that coefficients γ^L and γ^R can be transferred in an exact manner by a simple injection operator.

2.1.1. Restriction operator

Since such operator will act on the defect $\mathbf{r}_h = (\mathbf{r}_h^L, \mathbf{r}_h^R) \in \overset{\circ}{S}(\Omega_h)$ (step 4), we perform the restriction from a fine grid to a coarser grid separately for $\mathbf{r}_h^L$ and $\mathbf{r}_h^R$. This is justified by the fact that the defect $\mathbf{r}_h^L$ of the left domain may be very different (after few relaxations) from the defect $\mathbf{r}_h^R$ of the right domain, especially in the case of high jumping coefficient, i.e., $\max\left\{\gamma_\alpha^L/\gamma_\alpha^R, \gamma_\alpha^R/\gamma_\alpha^L\right\} \gg 1$. In addition, these defects are very different also from the defects of jumping conditions $\tilde{g}_D$ and $\tilde{g}_N$, because the operators scale with different power of h.

Let us describe the restriction of $\mathbf{r}_h^L$ by the operator $\left(I_{2h}^h\right)^L$ (see Fig. 2). Let x_J be the closest grid point to α from the left in the fine grid (see Fig. 1). Let x be a grid point of the coarse grid. If $x < x_J$ we will use the standard full-weighting restriction operator (FW):

$$\left(I_{2h}^h\right)^L \mathbf{r}_h^L(x) = \frac{1}{4}\left(\mathbf{r}_h(x-h)^L + 2\,\mathbf{r}_h(x)^L + \mathbf{r}_h(x+h)^L\right), \tag{2.10}$$

while if $x = x_J$ we reduce to an *upwind* linear convex combination from the left direction:

$$\left(I_{2h}^h\right)^L \mathbf{r}_h^L(x) = \omega_1\,\mathbf{r}_h^L(x) + (1-\omega_1)\mathbf{r}_h^L(x-h), \tag{2.11}$$

since in $x+h$ only $\mathbf{r}_h^R$ is defined and not $\mathbf{r}_h^L$. In our tests we found that $\omega_1 = 1/2$ gives better results than $\omega_1 = 3/4$.

The operator $\left(I_{2h}^h\right)^R$ works in a similar manner: let x_{J+1} the closest grid point to α from the right in the fine grid. If $x > x_{J+1}$ we will use the standard full-weighting restriction operator (FW):

$$\left(I_{2h}^h\right)^R \mathbf{r}_h^R(x) = \frac{1}{4}\left(\mathbf{r}_h(x-h)^R + 2\,\mathbf{r}_h(x)^R + \mathbf{r}_h(x+h)^R\right), \tag{2.12}$$

while if $x = x_J$ we reduce to an *Upwind* mean value from the left direction:

$$\left(I_{2h}^h\right)^R \mathbf{r}_h^R(x) = \frac{1}{2}\left(\mathbf{r}_h^R(x) + \mathbf{r}_h^R(x+h)\right). \tag{2.13}$$

The whole restriction reads

$$I_{2h}^h \mathbf{r}_h = \left(\left(I_{2h}^h\right)^L \mathbf{r}_h^L, \left(I_{2h}^h\right)^R \mathbf{r}_h^R\right).$$

In the upper part of Fig. 2 is represented the case in which we have to use (2.11) and (2.12). The only other possible case is that we have to use (2.10) and (2.13).

2.1.2. Interpolation operator

Since such operator will act on the correction $\mathbf{e}_{2h} = (\mathbf{e}_{2h}^L, \mathbf{e}_{2h}^R) \in S(\Omega_{2h})$ (step 4), we perform the interpolation from a coarse grid to a finer grid separately for $\mathbf{e}_{2h}^L$ and $\mathbf{e}_{2h}^R$ (see middle and lower part of Fig. 2), but always using the standard linear interpolation:

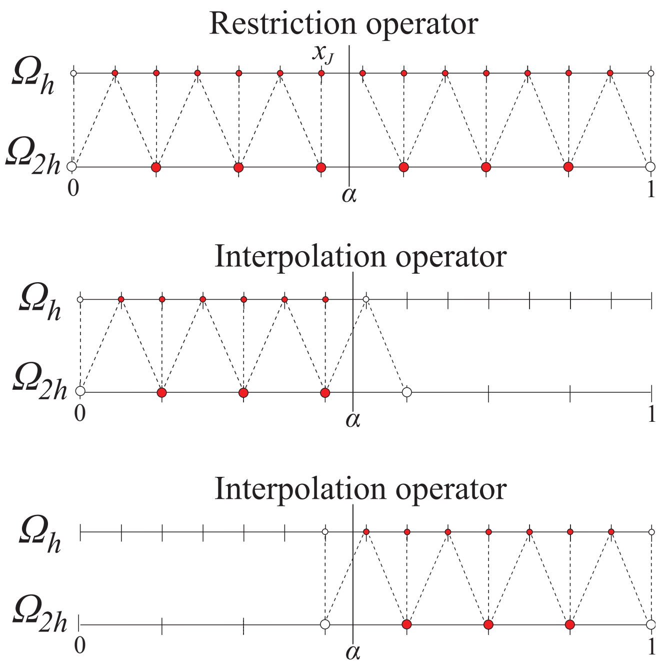


Figure 2: Fine and coarse grid for transfer operators. The dashed lines represent the action of the restriction (top) and the interpolation (middle and bottom) operators.

$$\begin{cases} \left(I_h^{2h}\right)^L \mathbf{e}_{2h}^L(x_j) = \mathbf{e}_{2h}^L(x_j) & \text{if } j \text{ is even,} \\ \left(I_h^{2h}\right)^L \mathbf{e}_{2h}^L(x_j) = \frac{1}{2}\left(\mathbf{e}_{2h}^L(x_{j-1}) + \mathbf{e}_{2h}^L(x_{j+1})\right) & \text{if } j \text{ is odd,} \end{cases} \tag{2.14}$$

$$\begin{cases} \left(I_h^{2h}\right)^R \mathbf{e}_{2h}^R(x_j) = \mathbf{e}_{2h}^R(x_j) & \text{if } j \text{ is even,} \\ \left(I_h^{2h}\right)^R \mathbf{e}_{2h}^R(x_j) = \frac{1}{2}\left(\mathbf{e}_{2h}^R(x_{j-1}) + \mathbf{e}_{2h}^R(x_{j+1})\right) & \text{if } j \text{ is odd.} \end{cases} \tag{2.15}$$

The whole interpolation reads

$$I_h^{2h}\mathbf{e}_{2h} = \left(\left(I_h^{2h}\right)^L \mathbf{e}_{2h}^L, \left(I_h^{2h}\right)^R \mathbf{e}_{2h}^R\right).$$

Remark 1. (Coarser operator) We observe that the discrete operator L_{2h} on the coarser grid (step 5) is just the operator obtained discretizing directly the continuous operator in the grid with spatial step $2h$, and not the operator obtained by Galerkin condition

$$L_{2h} = I_{2h}^h L_h I_h^{2h}. \tag{2.16}$$

The last approach, typical of algebraic multigrid, makes the algebraic problem more expensive from a computational point of view and does not take advantage of the fact that the discrete problem comes from a continuous problem.

Remark 2. (V-cycle) The V-cycle algorithm is easily obtained from the TGCS recursively, namely applying the same algorithm to solve the residual equation in step 5. To terminate the recursion, an exact solver is used to solve the residual problem when the grid achieves a fixed level of coarsening. We denote by $V(\nu_1,\nu_2)$-cycle the V-cycle performed with ν_1 pre-relaxations and ν_2 post-relaxations.

Remark 3. (W-cycle) The W-cycle is similar to the V-cycle, with the only difference that the residual problem is solved recursively two times instead of one (in general schemes, δ times, but $\delta > 2$ is considered useless for practical purposes).

3. Numerical tests

In this section we confirm numerically the second order accuracy of the discretization of Sec. 1.2 and compute the convergence factor ρ of the multigrid approach for several examples, to confirm the independence of ρ from the spatial step h and the magnitude of the jumping coefficient.

Second order accuracy is gained also for first derivative of the solution, as it is shown by the comparison between exact first derivative and the numerical derivative obtained by central difference of the numerical solution.

In all numerical tests, we choose an arbitrary interface $\alpha \in]0,1[$ and an analytical expression of the exact solution $u = (u^L, u^R)$ and of diffusion coefficient $\gamma = (\gamma^L, \gamma^R)$. Then we reconstruct the data f, g_D and g_N, perform the multigrid technique, and compare the numerical solution with the exact solution to compute the order of accuracy by the slope of the best-fit line. In all our tests we use the following stopping criterion for the $V-$cycle

$$\frac{\left\|\mathbf{u}_h^{(m+1)} - \mathbf{u}_h^{(m)}\right\|_\infty}{\left\|\mathbf{u}_h^{(m+1)}\right\|_\infty} \le TOL.$$

This will ensure that the actual relative error satisfies

$$\frac{\left\|\mathbf{e}_h^{(m+1)}\right\|_\infty}{\left\|\mathbf{e}_h\right\|_\infty} \le \rho \frac{TOL}{1-\rho}.$$

The tolerance we used is $TOL = 10^{-6}$, which ensures that the error in the solution of the algebraic system is always lower than truncation error. For each example we show a table in which we list the errors, and the value in the third [fifth] column and i-th row of the table indicates the accuracy order, computed as $\log_2\left(e_{i-1}/e_i\right)$, where e_i is the L^∞-error of the numerical solution [derivative] indicated in the second [fourth] column and i-th row.

To compute the asymptotic convergence factor, we use the following estimate:

$$\rho = \rho^{(m)} = \frac{\left\|\mathbf{r}_h^{(m)}\right\|_\infty}{\left\|\mathbf{r}_h^{(m-1)}\right\|_\infty},$$

which is reliable for m large. In order to avoid difficulties related to numerical instability due to machine precision, we will always use the homogeneous model problem as a test

Table 1: Measured $V(1,1)$-cycle convergence factor for the numerical test of Ex. 3.1. We use $N+2$ number of grid points in the finest grid; N_c+2 number of grid points in the coarsest grid.

N+1 / Nc+1	32	64	128	256	512	1024	2048	4096
16	0.15	0.16	0.15	0.17	0.19	0.15	0.15	0.15
32		0.14	0.12	0.17	0.19	0.15	0.15	0.15
64			0.07	0.16	0.19	0.15	0.15	0.15
128				0.11	0.17	0.15	0.15	0.15

when we want to compute the asymptotic convergence factor, namely Eq. (1.1) with $f = g_0 = g_1 = 0$ and homogeneous jump conditions, and perform the multigrid algorithm starting from an initial guess different from zero. Since in this case we are just interested in the convergence factor and not in the numerical solution itself (which approaches zero), a reasonable stop criterion will be

$$\frac{\left|\rho^{(m)} - \rho^{(m-1)}\right|}{\rho^{(m)}} < 10^{-2}.$$

Several tests are performed for each example, based on the different size of the finest and coarsest grids. The finest grid is obtained dividing the domain $[0,1]$ into $N+1$ intervals, while the coarsest grid is obtained dividing the domain into N_c+1 intervals.

Example 3.1. We choose (see Fig. 3)

$$\alpha = 0.343, \quad \begin{cases} u^L = e^{\sin(5\pi x)}, \\ u^R = e^{x^2}, \end{cases} \qquad \begin{cases} \gamma^L = 3 + \cos(5\pi x), \\ \gamma^R = 10^9\left(10 + \sin(5\pi x)\right). \end{cases}$$

Fig. 4 shows the numerical results and the second order slope of the best-fit line for the L^∞-error of the numerical solution and its derivative. Table 1 shows the convergence factor for different values of N and N_c.

Example 3.2. We choose (see Fig. 5)

$$\alpha = 0.743, \quad \begin{cases} u^L = e^{\sin(5\pi x)}, \\ u^R = e^{x^2}, \end{cases} \qquad \begin{cases} \gamma^L = 3 + \cos(5\pi x), \\ \gamma^R = 10^9\left(10 + \sin(5\pi x)\right). \end{cases}$$

The only difference with respect to the previous example is the value of α.

Fig. 6 shows the numerical results and the second order slope of the best-fit line for the L^∞-error of the numerical solution and its derivative. Table 2 shows the convergence factor for different values of N and N_c.

Example 3.3. We choose (see Fig. 7)

$$\alpha = 0.283, \quad \begin{cases} u^L = e^{\sin(5\pi x)}, \\ u^R = e^{x^2}, \end{cases} \qquad \begin{cases} \gamma^L = 10^9\left(10 + \sin(5\pi x)\right), \\ \gamma^R = 3 + \cos(5\pi x). \end{cases}$$

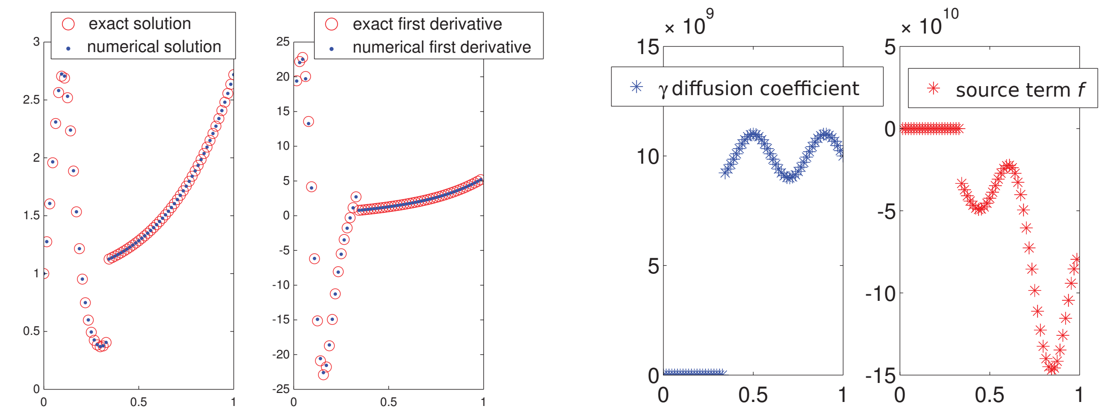
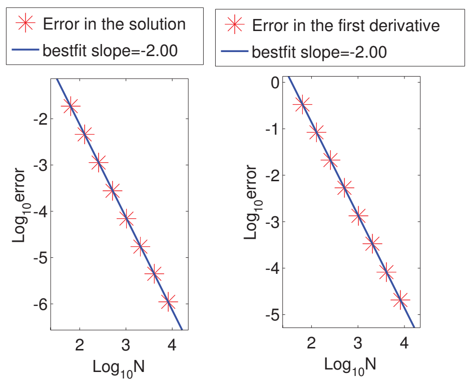


Figure 3: We refer to Ex. 3.1. The data are computed for $N = 64$.



$N+1$	$\|\mathbf{u}-\mathbf{u}_h\|_\infty$	order	$\|\mathbf{u}'-\mathbf{u}_h'\|_\infty$	order
64	$1.87 \cdot 10^{-2}$	-	$3.33 \cdot 10^{-1}$	-
128	$4.59 \cdot 10^{-3}$	2.03	$8.38 \cdot 10^{-2}$	1.99
256	$1.13 \cdot 10^{-3}$	2.02	$2.12 \cdot 10^{-2}$	1.98
512	$2.77 \cdot 10^{-4}$	2.03	$5.35 \cdot 10^{-3}$	1.99
1024	$6.95 \cdot 10^{-5}$	2.00	$1.34 \cdot 10^{-3}$	2.00
2048	$1.73 \cdot 10^{-5}$	2.01	$3.36 \cdot 10^{-4}$	1.99
4096	$4.48 \cdot 10^{-6}$	1.95	$8.21 \cdot 10^{-5}$	2.03
8192	$1.11 \cdot 10^{-6}$	2.01	$2.07 \cdot 10^{-5}$	1.99

Figure 4: We refer to Ex. 3.1. Left: Representation of the L^∞-error of the numerical solution and its derivative. The slope of the best-fit lines is respectively $s = -2.00$ and $s = -2.00$. Right: List of errors and order of accuracy computed by subsequent errors.

Table 2: Measured $V(1,1)$-cycle convergence factor for the numerical test of Ex. 3.2. We use $N+2$ number of grid points in the finest grid; N_c+2 number of grid points in the coarsest grid.

N+1	32	64	128	256	512	1024	2048	4096
Nc+1								
16	0.13	0.13	0.11	0.14	0.15	0.15	0.15	0.15
32		0.13	0.11	0.15	0.15	0.15	0.15	0.15
64			0.11	0.13	0.15	0.15	0.15	0.15
128				0.15	0.15	0.15	0.15	0.15

Fig. 8 shows the numerical results and the second order slope of the best-fit line for the L^∞-error of the numerical solution and its derivative. Table 3 shows the convergence factor for different values of N and N_c.

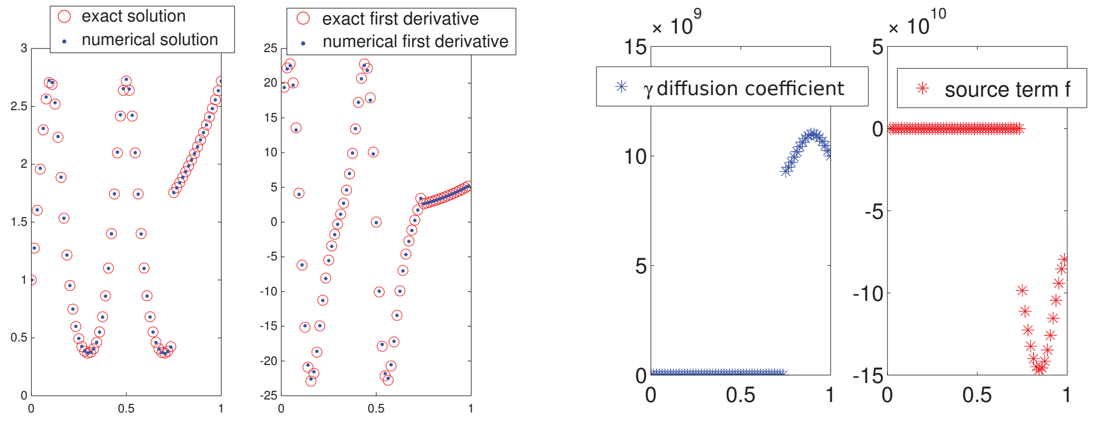
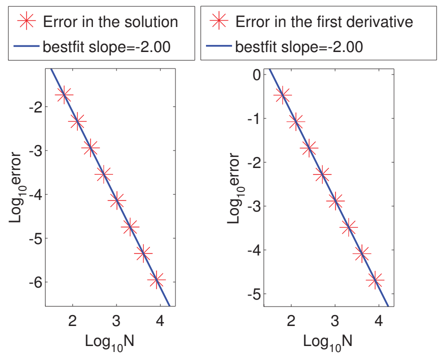


Figure 5: We refer to Ex. 3.2. The data are computed for $N=64$.



$N+1$	$\lVert\mathbf{u}-\mathbf{u}_h\rVert_\infty$	order	$\lVert\mathbf{u}'-\mathbf{u}'_h\rVert_\infty$	order
64	$1.86\cdot10^{-2}$	-	$3.38\cdot10^{-1}$	-
128	$4.63\cdot10^{-3}$	2.01	$8.38\cdot10^{-2}$	2.01
256	$1.15\cdot10^{-3}$	2.01	$2.10\cdot10^{-2}$	2.00
512	$2.86\cdot10^{-4}$	2.01	$5.26\cdot10^{-3}$	2.00
1024	$7.24\cdot10^{-5}$	1.98	$1.30\cdot10^{-3}$	2.01
2048	$1.80\cdot10^{-5}$	2.01	$3.28\cdot10^{-4}$	1.99
4096	$4.48\cdot10^{-6}$	2.01	$8.21\cdot10^{-5}$	2.00
8192	$1.12\cdot10^{-6}$	2.00	$2.05\cdot10^{-5}$	2.00

Figure 6: We refer to Ex. 3.2. Left: Representation of the L^∞-error of the numerical solution and its derivative. The slope of the best-fit lines is respectively $s=-2.00$ and $s=-2.00$. Right: List of errors and order of accuracy computed by subsequent errors.

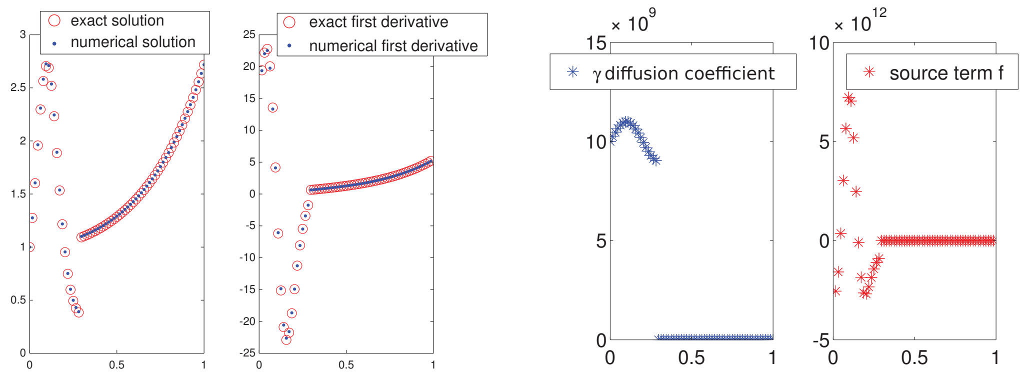
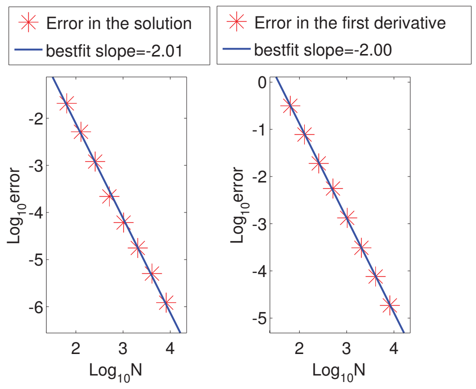


Figure 7: We refer to Ex. 3.3. The data are computed for $N=64$.

Example 3.4. We choose (see Fig. 9)

$$\alpha=0.813,\quad \begin{cases} u^L=e^{x^2},\\ u^R=e^{\sin(5\pi x)}, \end{cases}\qquad \begin{cases} \gamma^L=10^9\left(10+\sin(5\pi x)\right),\\ \gamma^R=3+\cos(5\pi x). \end{cases}$$



$N+1$	$\lVert \mathbf{u}-\mathbf{u}_h \rVert_\infty$	order	$\lVert \mathbf{u}'-\mathbf{u}_h' \rVert_\infty$	order
64	$2.07 \cdot 10^{-2}$	-	$3.15 \cdot 10^{-1}$	-
128	$5.15 \cdot 10^{-3}$	2.01	$7.76 \cdot 10^{-2}$	2.02
256	$1.20 \cdot 10^{-3}$	2.10	$1.90 \cdot 10^{-2}$	2.03
512	$2.19 \cdot 10^{-4}$	2.46	$5.57 \cdot 10^{-3}$	1.77
1024	$6.10 \cdot 10^{-5}$	1.84	$1.33 \cdot 10^{-3}$	2.07
2048	$1.76 \cdot 10^{-5}$	1.79	$3.09 \cdot 10^{-4}$	2.11
4096	$5.05 \cdot 10^{-6}$	1.80	$7.60 \cdot 10^{-5}$	2.02
8192	$1.22 \cdot 10^{-6}$	2.05	$1.86 \cdot 10^{-5}$	2.03

Figure 8: We refer to Ex. 3.3. Left: Representation of the L^∞-error of the numerical solution and its derivative. The slope of the best-fit lines is respectively $s=-2.01$ and $s=-2.00$. Right: List of errors and order of accuracy computed by subsequent errors.

Table 3: Measured $V(1,1)$-cycle convergence factor for the numerical test of Ex. 3.3. We use $N+2$ number of grid points in the finest grid; N_c+2 number of grid points in the coarsest grid.

N+1 / Nc+1	32	64	128	256	512	1024	2048	4096
16	0.09	0.10	0.12	0.15	0.15	0.15	0.15	0.15
32		0.09	0.10	0.15	0.15	0.15	0.15	0.15
64			0.12	0.15	0.15	0.15	0.15	0.15
128				0.13	0.15	0.15	0.15	0.15

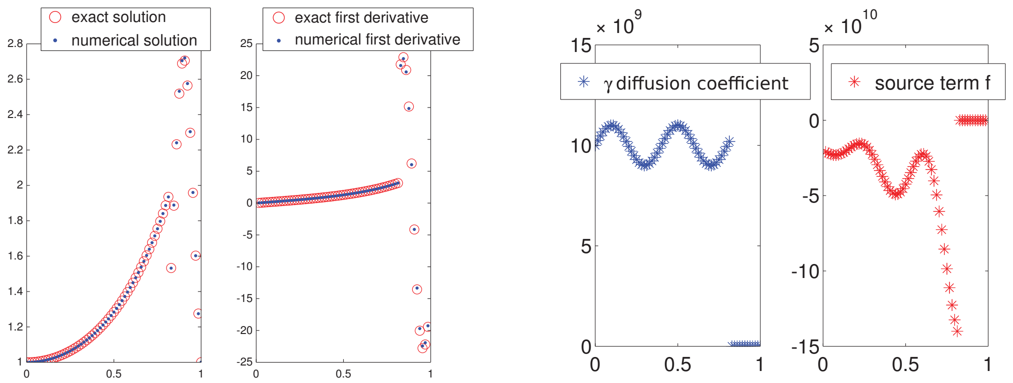
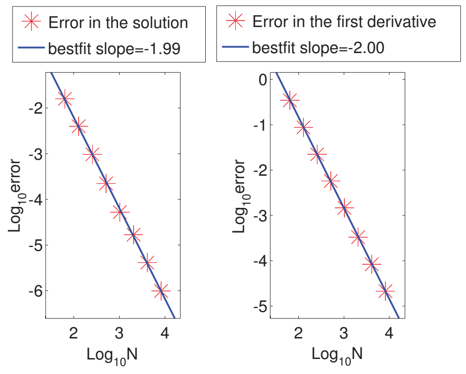


Figure 9: We refer to Ex. 3.4. The data are computed for $N=64$.

Fig. 10 shows the numerical results and the second order slope of the best-fit line for the L^∞-error of the numerical solution and its derivative. Table 4 shows the convergence factor for different values of N and N_c.



$N+1$	$\|\mathbf{u}-\mathbf{u}_h\|_\infty$	order	$\|\mathbf{u}'-\mathbf{u}_h'\|_\infty$	order
64	$1.59 \cdot 10^{-2}$	-	$3.46 \cdot 10^{-1}$	-
128	$3.99 \cdot 10^{-3}$	2.00	$8.74 \cdot 10^{-2}$	1.98
256	$9.66 \cdot 10^{-4}$	2.05	$2.22 \cdot 10^{-2}$	1.98
512	$2.23 \cdot 10^{-4}$	2.12	$5.74 \cdot 10^{-3}$	1.95
1024	$5.25 \cdot 10^{-5}$	2.08	$1.47 \cdot 10^{-3}$	1.97
2048	$1.68 \cdot 10^{-5}$	1.64	$3.31 \cdot 10^{-4}$	2.15
4096	$4.12 \cdot 10^{-6}$	2.03	$8.36 \cdot 10^{-5}$	1.98
8192	$9.87 \cdot 10^{-7}$	2.06	$2.13 \cdot 10^{-5}$	1.97

Figure 10: We refer to Ex. 3.4. Left: Representation of the L^∞-error of the numerical solution and its derivative. The slope of the best-fit lines is respectively $s=-1.99$ and $s=-2.00$. Right: List of errors and order of accuracy computed by subsequent errors.

Table 4: Measured $V(1,1)$-cycle convergence factor for the numerical test of Ex. 3.4. We use $N+2$ number of grid points in the finest grid; N_c+2 number of grid points in the coarsest grid.

N+1 / Nc+1	32	64	128	256	512	1024	2048	4096
16	0.17	0.12	0.14	0.18	0.17	0.15	0.16	0.15
32		0.11	0.14	0.16	0.15	0.15	0.15	0.15
64			0.06	0.14	0.15	0.15	0.15	0.15
128				0.12	0.15	0.15	0.15	0.15

Table 5: Measured $V(1,1)$ asymptotic convergence factors for a problem with a jumping coefficient of the order 10^p.

p	0	1	2	3	4	5
ρ	0.11	0.10	0.11	0.11	0.11	0.10

3.1. Independence of convergence factor from the jump in the coefficient

In this section we show that the convergence factor does not depend on the jump in the coefficient. We choose

$$\alpha = 0.543, \quad \begin{cases} u^L = 0, \\ u^R = 0, \end{cases} \qquad \begin{cases} \gamma^L = 10^p, \\ \gamma^R = 1, \end{cases}$$

and start the multigrid process with an initial guess different from zero, in order to compute the asymptotic convergence factor. We list the results in Table 5.

Remark.3.1. Comparison with Domain Decomposition Method
Domain Decomposition Method (DDM) is another iterative method to solve elliptic prob-

lems with discontinuous coefficient, based on solving iteratively the two subproblems

$$-\frac{\partial}{\partial x}\left(\gamma^L \frac{\partial u^{L,(m+1)}}{\partial x}\right) = f \quad \text{in } [0,\alpha[\,, \tag{3.1a}$$

$$u^{L,(m+1)}(0) = g_0, \tag{3.1b}$$

$$u^{L,(m+1)}(\alpha) = u^{R,(m)}(\alpha), \tag{3.1c}$$

and

$$-\frac{\partial}{\partial x}\left(\gamma^R \frac{\partial u^{R,(m+1)}}{\partial x}\right) = f \quad \text{in }]\alpha,1]\,, \tag{3.2a}$$

$$\gamma^R \frac{\partial u^{R,(m+1)}(\alpha)}{\partial x} = \gamma^L \frac{\partial u^{L,(m+1)}(\alpha)}{\partial x}, \tag{3.2b}$$

$$u^{R,(m+1)}(1) = g_1, \tag{3.2c}$$

until convergence. A little drawback of this method is that, in order to guarantee the convergence, it must be $\alpha > 0.5$ (see [27, pag. 12]). Our method may be regarded as a DDM, but in place of solving a subproblem to provide the right-hand side for the other subproblem (and so on iteratively), we just perform a relaxation on a subproblem, and with the guess obtained we build the right-hand side of the other subproblem, as it can be seen in Sec. 1.3. With this relaxing strategy, the convergence is always guaranteed, as showed in numerical tests.

Conclusion

A second order discretization for elliptic equation with discontinuous coefficient on an arbitrary interface has been provided. Second order accuracy in the derivative is obtained as well. The linear system is solved by an iterative method obtained relaxing the interface conditions. The iterative method is then speeded up by a proper multigrid approach, which transfers separately the defect for both sub-problems obtained from the multi-domain formulation. The measured convergence factor is close to the one measured in the case of smooth coefficients and it does not depend on the magnitude of the jump in the coefficient. The method is similar to Domain Decomposition Methods, but a single relaxation sweep is performed in each subdomain instead to solve it completely. This makes the method more flexible and there is no restriction on the relative size of the two subdomains.

This paper is the building-block for a future work in higher dimension [12], which will be carried out by combining the second order discretization in arbitrary domain with smooth coefficients [10] and the multigrid treatment of problems with non-eliminated boundary conditions in arbitrary domain [11].

Other future works concern the convection-diffusion equation in a moving domain, in order to study applications modeled by a Stefan-Type problem. A level-set function will keep track of the moving interface.

All this extensions will be coupled with the use of Adaptive Mesh Refinement to obtain accurate solution in the case of domain with complex boundary. A proper multigrid approach is under investigation for all these works.

References

[1] R. E. Alcouffe, A. Brandt, J. Dendy, J. E., and J. W. Painter. The multigrid method for the diffusion equation with strongly discontinuous coefficients. *Journal on Scientific and Statistical Computing*, 2:430–454, 1981.

[2] P. Angot, C.-H. Bruneau, and P. Fabrie. A penalization method to take into account obstacles in incompressible viscous flows. *Numer. Math.*, 81, 1999.

[3] I. Babuška. The finite element method for elliptic equations with discontinuous coefficients. *Computing*, 5:207–213, 1970.

[4] G. Bao, G. Wei, and S. Zhao. Numerical solution of the Helmholtz equation with high wave numbers. *J. Numer. Methods Engng.*, 59:389–408, 2004.

[5] J. Bramble and J. King. A finite element method for interface problems in domains with smooth boundaries and interfaces. *Adv. Comput. Math.*, 6:109–138, 1996.

[6] J. H. Bramble and B. E. Hubbard. Approximation of solutions of mixed boundary value problems for Poisson's equation by finite differences. *J. Assoc. Comput. Mach.*, 12:114–123, 1965.

[7] W. L. Briggs, V. E. Henson, and S. F. McCormick. *A Multigrid Tutorial*. SIAM, 2000.

[8] F. Chantalat, C.-H. Bruneau, C. Galusinski, and A. Iollo. Level-set, penalization and cartesian meshes: A paradigm for inverse problems and optimal design. *Journal of Computational Physics*, 228:6291–6315, 2009.

[9] H. Chen, C. Min, and F. Gibou. A supra-convergent finite difference scheme for the Poisson and heat equations on irregular domains and non-graded adaptive Cartesian grids. *Journal of Scientific Computing*, 31:19–60, 2007.

[10] A. Coco and G. Russo. A fictitious time method for the solution of Poisson equation in an arbitrary domain embedded in a square grid. *Journal of Computation Physics*. Under revision.

[11] A. Coco and G. Russo. Multigrid approach for Poisson's equation with mixed boundary condition in an arbitrary domain.

[12] A. Coco and G. Russo. Second order multigrid methods for elliptic problems with discontinuous coefficients on an arbitrary interface, II: higher dimensional problems.

[13] J. Donea. An arbitrary Lagrangian-Eulerian finite element method for transient fluid-structure interactions. *Computer Methods in Applied Mechanics and Engineering*, 33:689–723, 1982.

[14] R. Fedkiw, T. Aslam, B. Merriman, and S. Osher. A Non-Oscillatory Eulerian Approach to Interfaces in Multimaterial Flows (The Ghost Fluid Method). *Journal of Computational Physics*, 152:457–492, 1999.

[15] L. Formaggia and F. Nobile. Stability analysis of second-order time accurate schemes for ALE-FEM. *Computer Methods in Applied Mechanics and Engineering*, 193:4097–4116, 2004.

[16] F. Gibou and R. Fedkiw. A second-order-accurate symmetric discratization of the poisson equation on irregular domains. *Journal of Computational Physics*, 176:205–227, 2002.

[17] F. Gibou and R. Fedkiw. A fourth order accurate discretization for the laplace and heat equations on arbitary domains, with applications to the stefan problem. *Journal of Computational Physics*, 202:577–601, 2005.

[18] R. Glowinski, T. W. Pan, T. I. Hesla, and D. D. Joseph. A distributed Lagrange multiplier/fictitious domain method for particulate flows. *International Journal of Multiphase Flow*,

25:755–794, 1999.
[19] W. Hackbusch. *Multi-grid methods and applications*. Springer, 1985.
[20] W. Hackbusch. *Elliptic Differential Equations: Theory and Numerical Treatment*. Springer, 2003.
[21] J. J. E. Dendy. Black Box Multigrid. *Journal of Computational Physics*, 48:366–386, 1982.
[22] H. Johansen and P. Colella. A Cartesian Grid Embedded Boundary Method for Poisson Equation on Irregular Domains. *Journal of Computational Physics*, 147:60–85, 1998.
[23] R. LeVeque and Z. Li. The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. *SIAM J. Numer. Anal.*, 31:1019–1044, 1994.
[24] A. Mayo. The fast solution of Poisson's and the biharmonic equations on irregular regions. *SIAM J. Numer. Anal.*, 21:285–299, 1984.
[25] J. Papac, F. Gibou, and C. Ratsch. Efficient Symmetric Discretization for the Poisson, Heat and Stefan-Type Problems with Robin Boundary Conditions. *Journal of Computational Physics*, 229:875–889, 2010.
[26] C. S. Peskin. Numerical analysis of blood flow in the heart. *Journal of Computational Physics*, 25:220–252, 1977.
[27] A. Quarteroni and A. Valli. *Domain Decomposition Methods for Partial Differential Equations*. Numerical Mathematics and Scientific Computation, 1999.
[28] J. W. Ruge and Stüben. *Algebraic multigrid. Multigrid methods*.
[29] A. Sarthou, S. Vincent, J. Caltagirone, and P. Angot. Eulerian-Lagrangian grid coupling and penalty methods for the simulation of multiphase flows interacting with complex objects. *International Journal for Numerical Methods in Fluids*, 00:1–6, 2007.
[30] G. H. Shortley and R. Weller. The numerical solution of laplace's equation. *J. Appl. Phys.*, 9:334–348, 1938.
[31] U.Trottemberg, C. Oosterlee, and A. Schuller. *Multigrid*. Academic Press, 2000.
[32] S. Yu, Y. Zhou, and G. Wei. Matched Interface and Boundary (MIB) method for elliptic problems with sharp-edged interfaces. *Journal of Computational Physics*, 224:729–756, 2007.