A Finite Difference Ghost-cell Multigrid approach for Poisson Equation with mixed Boundary Conditions in Arbitrary Domain

Armando Coco, Giovanni Russo*

November 7, 2011

Abstract

In this paper we present a multigrid approach to accelerate the convergence of the iterative method proposed in [13] to solve the Poisson equation in arbitrary domain Ω , identified by a level set function φ , $\Omega = \left\{x \in \mathbb{R}^d : \varphi(x) < 0\right\}$, and mixed boundary conditions. The discretization is based on finite difference scheme and ghost-cell method. This multigrid strategy can be applied also to more general problems where a non-eliminated boundary condition approach is used. Arbitrary domain make the definition of the restriction operator for boundary conditions hard to find. A suitable restriction operator is provided in this work, together with a proper treatment of the boundary smoothing, in order to avoid degradation of the convergence factor of the multigrid due to boundary effects. Several numerical tests confirm the good convergence property of the new method.

Introduction

Multigrid technique is one of the most efficient strategy to solve a class of partial differential equations, using a hierarchy of discretizations. It accelerates the convergence of an existing iterative method, which otherwise slowly converges toward the solution of the discrete problem, due to the bad convergence rate for the low frequency components of the error. The idea of multigrid method is to solve such low frequency component in a coarser grid. An introduction to multigrid can be found, for example, in [4], while more advanced textbook on the subject are, for example, [41, 21]. Most iterative schemes to solve Elliptic equations can be speeded up by a multigrid technique.

Elliptic equation in arbitrary domain (possibly with moving boundary) is central to many applications, such as diffusion phenomena, fluid dynamics, charge transport in semiconductors, crystal growth, electromagnetism and many others. The wide range of applications may require different kind of boundary conditions. Let us look for instance at the temperature distribution in a medium of arbitrary shape satisfying stationary heat equation: we may have Dirichlet (the temperature is fixed at the boundary), Neumann (heat flux is prescribed), or mixed boundary conditions (namely different boundary conditions on different

^{*}Dipartimento di Matematica e Informatica, Università di Catania, Catania Italy

part of the boundary). More general Robin boundary conditions may also be prescribed, as in Stefan-type problem, in which a combination of temperature and heat flux is prescribed at the boundary (e.g. see [20, 5]). An application we have in mind is to fluid dynamics: the aim is to model the motion of an incompressible fluid contained in a tank of arbitrary shape. The problem is modeled by incompressible Navier-Stokes equations, which are solved by projection method of Chorin [10, 11]. This leads to an elliptic equation for the pressure, obtained enforcing the incompressibility condition. This pressure equation requires Dirichlet condition on the free surface of the fluid and Neumann condition on the rigid walls. The pressure equation is the bottleneck of the whole method and therefore requires an efficient solver.

Several techniques have been developed to solve Elliptic equation on an arbitrary domain. Finite Element Methods use a mesh triangulation to capture the boundary, such as in [34, 35, 30, 37]. However, in presence of moving boundary, a grid re-meshing is needed at each time step, which makes the method expensive. Furthermore, for a complex geometry, generation of a good mesh is a non trivial task that may require a considerable amount of work [28]. For this reason a Cartesian grid method is preferred together with a level-set approach to keep track of the boundary at each time step. Level-set methods have been introduced to implicitly define a domain and its boundary, in order to simple handle complex topological changes of moving boundary such as merging and breaking up. Several papers and books exist in the literature about level-set method: [40, 31, 36, 17, 25] are just some examples.

Since the boundary may be not aligned with the grid, a special treatment is needed. The simplest method makes use of the Shortley-Weller discretization [38], that discretizes the Laplacian operator with usual central difference away from the boundary and makes use of a non symmetric stencil in the interior points of the domain close to the boundary. While this discretization provides a simple second order method for Dirichlet conditions, it cannot be immediately applied in presence of Neumann conditions. In fact, Shortley-Weller discretization [38] for Neumann conditions requires that the value of the numerical solution is suitably reconstructed at the intersection between the grid and the boundary by applying the boundary condition. This approach is adopted, for example, by Hackbusch in [22] to first order accuracy, and by other authors (see [2] and the references therein) to second order accuracy. However, the method proposed by Bramble in [2] for second order accuracy is quite involved and not recommendable for practical purposes.

Another class of methods is based on cut-cell methods, obtained by a Finite Volume discretization which embeds the domain in a regular Cartesian grid [24]. 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 (e.g. see [23, 9, 12]).

Other methods for Dirichlet condition are the Immersed Boundary Method, first proposed by Peskin in [33], and later developed by several other authors [27, 42], with a proper multigrid approach [1], and penalization methods [7].

In our method we will use a rather simple finite-difference ghost-cell technique, that adds 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 boundary. A detailed description of the method can be found in [13].

In ghost points the boundary conditions are enforced in order to close the discrete system. The ghost-cell method was first developed by Fedkiw in [18], where a two-phase contact discontinuity was discretized

(Ghost Fluid Method). A second-order accurate method for Dirichlet conditions on regular Cartesian grid is proposed by Gibou et al. in [19]. 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 [20]. Other methods use a non-regular Cartesian grid, such as in [8], 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 [32], which provides second order accuracy for the Poisson and heat equation and first order accuracy for Stefan-type problems.

Most of the techniques listed above cannot be straightforwardly applied in the special case of mixed boundary conditions. For cut-cell based methods [24, 32], different boundary conditions cannot to be imposed on the same boundary edge of a cut cell. Simple efficient methods based on symmetric image of ghost points to solve mixed boundary condition problems provided with a multigrid algorithm have been recently developed in [6] and by Ma et al. in [29].

In our method [13], boundary 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), which is proved to converge, at least for first order accurate discretization.

In this paper we provide a general multigrid technique to solve the discrete system coming from a continuous elliptic problem in case of non-eliminated boundary conditions. The smoothing procedure of the multigrid approach in the interior is Gauss-Seidel-like, while the iterations on the boundary are performed in order to provide smooth errors.

Multigrid techniques for non-eliminated boundary conditions are well-studied in literature in the case of rectangular domain (as we can see in [21, 41]), where a restriction operator is defined separately for the interior of the domain and for the boundary, and the restriction of the boundary is performed using a restriction operator of codimension 1, since ghost points are aligned with the Cartesian axis. In the case of arbitrary domain, ghost points have an irregular structure and we provide a reasonable definition of the restriction operator for the boundary conditions. The method proposed in this paper can be extended to the case of discontinuous coefficient: a preliminary result in one dimension can be found in [14], while the two-dimensional case is in preparation.

In this paper we also show that a proper treatment of the boundary iterations can improve the rate of convergence of the multigrid, making it closer to the convergence rate predicted by the Local Fourier Analysis for inside equations, as suggested by Brandt in [3]. The cost of this extra computational work is negligible, i.e. tends to zero as the dimension of the problem increases. A comparison with other kinds of treatment of the boundary condition smoothing procedure is carried out.

The paper is divided in three sections. We start with the multigrid approach in the one dimensional case, described in Section 1, with a special treatment of the transfer operators. Most of this method can be extended to high dimension, treated in Section 2, but a special care has to be taken for transferring the defect of boundary conditions. In this section a level-set approach is also introduced. Section 3 provides a strategy

to improve the convergence factor making it closer to the one predicted by the Local Fourier Analysis for interior relaxations. Numerical evidence of such improvement is provided, together with a comparison with other boundary condition smoothers (Kaczmarz and Block relaxation).

In all the paper, we mainly use the notation of [41].

One-dimensional case 1

In this section we will follow the description of the method proposed in [13], which is second order accurate, and provides a multigrid approach to speed up the convergence of the iterative scheme. For one-dimensional case, the multigrid approach in arbitrary domain is a natural extension of the basic multigrid strategy that can be found in any good basic text about multigrid, such as [41, 4, 21]. Although if we can eliminate the boundary conditions from the linear system obtained by discretizing the problem, we always want to treat the case of non-eliminated boundary conditions in order to straightforwardly extend the method to more than one dimension, where the elimination of the boundary conditions from the system is hard to perform.

Model problem and relaxation scheme 1.1

Let D = [-1, 1] be the computational domain, a and b constants such that -1 < a < b < 1, and $\Omega = [a, b]$. Let $N \ge 1$ be a fixed integer and h = 2/N the spatial step, let $D_h = \{-1 = x_0 < x_1 < \ldots < x_N = 1\}$ be the set of equally spaced grid points, and $\Omega_h = D_h \cap \Omega$ the set of inside grid points. Consider the model problem

$$-u'' = f \quad \text{in } \Omega$$

$$u(a) = g_a$$

$$u'(b) = g_b.$$
(1)

Let l and r be such that $x_l \leq a < x_{l+1}, x_{r-1} < b \leq x_r$ (see Figure 1). We use a ghost-cell method to discretize the problem. In order to obtain an iterative method, we solve the associate time-dependent problem

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + f \quad \text{in } \Omega \tag{2}$$

$$\frac{\partial u(t,a)}{\partial t} = \mu_D \left(g_a - u(t,a) \right) \tag{3}$$

$$\frac{\partial u(t,a)}{\partial t} = \mu_D (g_a - u(t,a))$$

$$\frac{\partial u(t,b)}{\partial t} = \mu_N \left(g_b - \frac{\partial u(t,b)}{\partial x} \right)$$
(3)

$$u(0,x) = u_0(x) \quad \text{in } \Omega \tag{5}$$

and we look for a second order accurate steady state solution, which is the solution of the original problem. Let us begin to discretize (2) in $\Omega_h \equiv \{x_{l+1}, \dots, x_{r-1}\}$. We use central difference in space and forward Euler in time for (2) obtaining:

$$u_i^{(m+1)} = u_i^{(m)} + \frac{\Delta t}{h^2} \left(u_{i-1}^{(m)} - 2u_i^{(m)} + u_{i+1}^{(m)} \right) + \Delta t f_i, \quad i = l+1, \dots, r-1.$$
 (6)

Taking the maximum time step consented by CFL condition [15], i.e. $\Delta t = h^2/2$, we obtain:

$$u_i^{(m+1)} = \frac{1}{2} \left(u_{i-1}^{(m)} + u_{i+1}^{(m)} + h^2 f_i \right), \quad i = l+1, \dots, r-1.$$
 (7)

Note that if we discretize directly the first equation (1) using central difference for the Laplacian operator, and use Jacobi iterative scheme for such discretization, we obtain exactly (7).

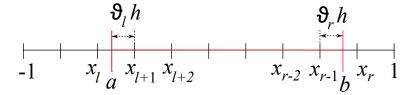


Fig. 1: Discretization of the domain in 1D

To obtain second order accuracy, we have to discretize the spatial terms in (3), (4) to second order, while first order time discretization can be used, because we are just interested at the accuracy as $t \to +\infty$.

We then can use linear interpolation for u(t, a) in (3). Since in some application it is required second order accuracy of the gradient of the solution, we use quadratic interpolation instead linear interpolation, obtaining:

$$u_l^{(m+1)} = u_l^{(m)} - \mu_D \Delta t \left((1 + \vartheta_l) \frac{\vartheta_l}{2} u_l^{(m)} + (1 + \vartheta_l) (1 - \vartheta_l) u_{l+1}^{(m)} - (1 - \vartheta_l) \frac{\vartheta_l}{2} u_{l+2}^{(m)} - g_a \right), \tag{8}$$

where $\vartheta_l = (x_{l+1} - a)/h$, and quadratic interpolation of u in nodes x_r, x_{r-1}, x_{r-2} for the $\partial u(t, b)/\partial x$ in (4), obtaining:

$$u_r^{(m+1)} = u_r^{(m)} - \frac{\mu_N \Delta t}{h} \left(u_{r-1}^{(m)} - u_{r-2}^{(m)} + \left(u_{r-2}^{(m)} - 2u_{r-1}^{(m)} + u_r^{(m)} \right) \left(\frac{1}{2} + \vartheta_r \right) \right) + \mu_N \Delta t g_b. \tag{9}$$

The constants μ_D and μ_N are chosen in order to satisfy the CFL conditions, i.e. $\mu_D \Delta t < 1$ and $\mu_N \Delta t/h < 2/3$ (see [13]). In numerical tests of Section 3, we choose $\mu_D \Delta t = 0.9$ and $\mu_N \Delta t = 0.9 \cdot 2h/(3)$. Since $\Delta t = h^2/2$, then $\mu_D = 1.8/h^2$ and $\mu_N = 3.6/(3h)$.

In summary, our second order accurate iterative method is described by Eqs. (7), (8) and (9), with the choice of constants

$$\Delta t = h^2/2, \quad \mu_D = 1.8/h^2, \quad \mu_N = 1.2/h.$$
 (10)

1.2 Multigrid approach

We call Γ_h the set of ghost points, i.e. $\Gamma_h = \{x_l, x_r\}$. Let I_h be a general subset of D_h . We introduce the linear space of grid functions over I_h and we denote it $S(I_h) = \{\mathbf{w}_h : I_h \to \mathbb{R}\}$. For any $\mathbf{w}_h \in S(I_h)$, we

pose $w_i^h = \mathbf{w}_h(x_i)$. Let $\mathbf{f}_h \in S(\Omega_h)$ such that $f_i^h = f(x_i)$. The iterative scheme (7), (8), (9) converges to the exact solution of the discretized system

$$-\Delta_h \mathbf{u}_h = \mathbf{f}_h \tag{11}$$

$$g_D^h(\mathbf{u}_h) = g_a \tag{12}$$

$$g_N^h(\mathbf{u}_h) = g_b, \tag{13}$$

where $\Delta_h \colon S(\Omega_h \cup \Gamma_h) \to S(\Omega_h)$ is defined by:

$$\Delta_h \mathbf{u}_h(x_i) = \frac{u_{i-1}^h - 2u_i^h + u_{i+1}^h}{h^2}, \quad x_i \in \Omega_h,$$

while $g_D^h, g_N^h \colon S(\Omega_h \cup \Gamma_h) \to \mathbb{R}$ are the discrete versions of the boundary conditions:

$$g_D^h(\mathbf{u}_h) = (1 + \vartheta_l) \frac{\vartheta_l}{2} u_l^h + (1 + \vartheta_l)(1 - \vartheta_l) u_{l+1}^h - (1 - \vartheta_l) \frac{\vartheta_l}{2} u_{l+2}^h,$$

$$g_N^h(\mathbf{u}_h) = \frac{u_{r-1}^h - u_{r-2}^h}{h} + \frac{u_{r-2}^h - 2u_{r-1}^h + u_r^h}{h} \left(\frac{1}{2} + \vartheta_r\right).$$

System (11)-(13) can be interpreted in general as a discrete system of a Poisson equation with non-eliminated boundary conditions.

Let us consider an arbitrary grid function $\tilde{\mathbf{u}}^h \in S(\Omega_h \cup \Gamma_h)$ and let

$$\mathbf{r}_h = \mathbf{f}_h + \Delta_h \tilde{\mathbf{u}}^h$$

$$\tilde{g}_a = g_a - g_D^h(\tilde{\mathbf{u}}^h)$$

$$\tilde{g}_b = g_b - g_N^h(\tilde{\mathbf{u}}^h)$$

be the defects of (11), (12), (13) respectively. Because of the linearity of Δ_h , g_D^h , g_N^h , if we solve exactly the so-called residual problem

$$-\Delta_h \mathbf{e}_h = \mathbf{r}_h \tag{14}$$

$$g_D^h(\mathbf{e}_h) = \tilde{g}_a \tag{15}$$

$$g_N^h(\mathbf{e}_h) = \tilde{g}_b \tag{16}$$

in the unknown $\mathbf{e}_h \in S(\Omega_h \cup \Gamma_h)$, then $\mathbf{u}_h = \tilde{\mathbf{u}}^h + \mathbf{e}_h$ is the exact solution of the system (11), (12), (13). In the basic idea of multigrid one needs to solve the residual problem in a grid coarser than the original one. We can summarize the iterative scheme (7), (8), (9) as follows:

$$\mathbf{u}_h^{(m+1)} = \Re_h \left(\mathbf{u}_h^{(m)}, \mathbf{f}_h, g_a, g_b \right) \tag{17}$$

$$\Re_h \colon S(\Omega_h \cup \Gamma_h) \times S(\Omega_h) \times \mathbb{R}^2 \longrightarrow S(\Omega_h \cup \Gamma_h).$$
 (18)

Note that the iterative scheme (7), (8), (9) is of a Jacobi kind. In order to provide a multigrid strategy, we just require that the iteration operator (18) has the *smoothing property*, i.e. after few iteration steps (17), the error becomes smooth (not necessarily small). Roughly speaking, the high-frequency components of the error reduce quickly. We call *smoothers* any operator (18) with this property. Many iterators have this property, such as Gauss-Seidel or weighted Jacobi (with weight $\omega = 2/3$ in 1D or $\omega = 4/5$ in 2D), but not Jacobi (see [41, pag. 30–32] for more details). From now on, by (17) we shall intend the Gauss-Seidel version of (7), (8), (9), i.e.:

$$u_{l}^{(m+1)} = u_{l}^{(m)} - \mu_{D} \Delta t \left(\vartheta_{l} u_{l}^{(m)} + (1 - \vartheta_{l}) u_{l+1}^{(m)} - g_{a} \right)$$

$$u_{i}^{(m+1)} = \frac{1}{2} \left(u_{i-1}^{(m+1)} + u_{i+1}^{(m)} + h^{2} f_{i} \right), \quad i = l+1, \dots, r-1$$

$$u_{r}^{(m+1)} = u_{r}^{(m)} + \mu_{N} \Delta t g_{b}$$

$$-\frac{\mu_{N} \Delta t}{h} \left(u_{r-1}^{(m+1)} - u_{r-2}^{(m+1)} + \left(u_{r-2}^{(m+1)} - 2 u_{r-1}^{(m+1)} + u_{r}^{(m+1)} \right) \left(\frac{1}{2} + \vartheta_{r} \right) \right).$$

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 [41, Sections 2.4, 2.6] for more details). The TGCS consists into the following algorithm:

- 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

$$\mathbf{u}_h := \Re_h \left(\mathbf{u}_h, \mathbf{f}_h, g_a, g_b \right)$$

3. Compute the defects

$$\mathbf{r}_h = \mathbf{f}_h + \Delta_h \mathbf{u}_h$$

$$\tilde{g}_a = g_a - g_D^h(\tilde{\mathbf{u}}^h)$$

$$\tilde{g}_b = g_b - g_N^h(\tilde{\mathbf{u}}^h)$$

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

$$-\Delta_{2h}\mathbf{e}_{2h} = \mathbf{r}_{2h} \tag{19}$$

$$g_D^{2h}(\mathbf{e}_{2h}) = \tilde{g}_a \tag{20}$$

$$g_N^{2h}(\mathbf{e}_{2h}) = \tilde{g}_b \tag{21}$$

in the unknown $\mathbf{e}_{2h} \in S(\Omega_{2h} \cup \Gamma_{2h})$

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

$$\mathbf{u}_h := \Re_h \left(\mathbf{u}_h, \mathbf{f}_h, g_a, g_b \right)$$

We have just to explain the steps concerning grid migration (steps 4 and 6).

1.3 Transfer grid operators

In this section, we describe the transfer grid operators for vertex-centered grid. We observe that our approach is based on the discretization of the equations on the various grids (both for inner and ghost points). This approach is very different from algebraic multigrid. As a consequence, the interpolation and the restriction operators are not the transpose of each other.

1.3.1 Restriction operator

Since such operator will act on the defect $\mathbf{r}_h \in S(\Omega_h)$ (step 4), we must determine $I_{2h}^h \mathbf{r}_h(x)$ for any $x \in \Omega_{2h}$ using only values inside Ω_h . This is justified by the fact that the defect of the inside grid points (referred to the Poisson equation) may be very different (after few relaxations) from the defects \tilde{g}_a , \tilde{g}_b (referred to the boundary conditions and stored computationally in the ghost points), because the operators (for inner equations and for boundary conditions) scale with different powers of h. Then, let $x \in \Omega_{2h}$ and refer to Fig. 2 (upper part). If x is not near an outside grid point, i.e. $\min\{|x-a|, |x-b|\} \ge h$, then we will use the standard full-weighting restriction operator (FW):

$$I_{2h}^{h}\mathbf{r}_{h}(x) = \frac{1}{4} \left(\mathbf{r}_{h}(x-h) + 2\mathbf{r}_{h}(x) + \mathbf{r}_{h}(x+h) \right),$$
 (22)

while if x - h < a or x + h > b we set respectively

$$I_{2h}^{h}\mathbf{r}_{h}(x) = \frac{1}{2}\left(\mathbf{r}_{h}(x) + \mathbf{r}_{h}(x+h)\right)$$
(23)

or

$$I_{2h}^{h}\mathbf{r}_{h}(x) = \frac{1}{2}\left(\mathbf{r}_{h}(x-h) + \mathbf{r}_{h}(x)\right). \tag{24}$$

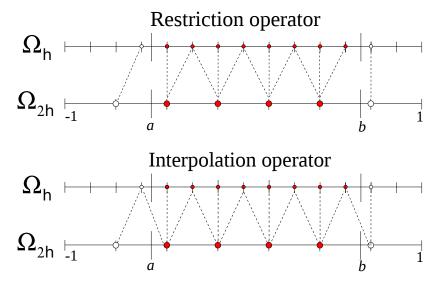


Fig. 2: Vertex-centered discretization in 1D. Inner grid nodes (red circles) and ghost points (empty circles) on the fine and coarse mesh. The dashed lines represent the action of the restriction (up) and the interpolation (down) operators.

1.3.2 Interpolation operator

Since the interpolation operator acts on the error (step 6), which is continuous across the boundary, we do not need to separate the interpolation for inner equations from the interpolation of ghost points, and then we just use the standard linear interpolation operator (see the lower part of Fig. 2):

$$I_h^{2h} \mathbf{e}_{2h}(x_j) = \mathbf{e}_{2h}(x_j)$$
 if j is even
$$I_h^{2h} \mathbf{e}_{2h}(x_j) = \frac{1}{2} \left(\mathbf{e}_{2h}(x_{j-1}) + \mathbf{e}_{2h}(x_{j+1}) \right)$$
 if j is odd.

Remark. 1 (*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 becomes too coarse.

Remark. 2 (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, γ times, but $\gamma > 2$ is considered useless for practical purpose).

Remark. 3 (Coarser operator) We observe that the discrete operator Δ_{2h} in step 5 is just the operator obtained discretizing directly the continuous operator in the coarser grid, and not the operator obtained by the Galerkin condition

$$\Delta_{2h} = I_{2h}^h \, \Delta_h \, I_h^{2h}.$$

The latter 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.

$\mathbf{2}$ High-dimensional case

In this case the defect of the boundary conditions has to be transferred in a suitable way to a coarser grid. The restriction has to be performed separately from the restriction of the interior equations, since these defects may show a sharp gradient crossing the boundary, because the discrete operators scale with different powers of h.

In case of arbitrary domain, ghost points may have a complex structure and the restriction cannot be defined straightforwardly as in the rectangular case, where ghost points are aligned with the grid and the restriction can be performed by a one dimensional operator.

For arbitrary domain we first need to extend the defect in a narrow band outside the domain constant along normal direction, and then we can operate the restriction as in the interior of the domain. For the sake of clarity, we describe the multigrid strategy in the two-dimensional case, but the procedure can be extended straightforwardly in more dimensions. We always refer to the second order method proposed in [13], which is briefly recalled here.

2.1Model problem and relaxation scheme

Let $D = [-1, 1]^2$ be the computational domain, $\Omega \subset D$ be a domain such that $\partial \Omega \cap \partial D = \emptyset$. Let Γ_D, Γ_N be a partition of $\partial\Omega$ (i.e. $\Gamma_D \cup \Gamma_N = \partial\Omega$, $\overset{\circ}{\Gamma}_D \cap \overset{\circ}{\Gamma}_N = \emptyset$, where the interior points are computed in the d-1dimensional topological space). Consider the model problem

$$-\Delta u = f \quad \text{in } \Omega
 u = g_D \quad \text{on } \Gamma_D
 \frac{\partial u}{\partial n} = g_N \quad \text{on } \Gamma_N,$$
(25)

where **n** is the outward unit normal, $\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$ is the Laplacian operator, $f: \Omega \to \mathbb{R}, g_D: \Gamma_D \to \mathbb{R}$, $g_N \colon \Gamma_N \to \mathbb{R}$ are assigned functions.

In order to solve the elliptic problem (25), we can transform it in an evolutive problem (with a fictitious time) that we call the associate time-dependent problem:

$$\frac{\partial \tilde{u}}{\partial t} = \Delta \tilde{u} + f \qquad \text{in } \Omega \tag{26}$$

$$\frac{\partial \tilde{u}}{\partial t} = \mu_D(g_D - \tilde{u}) \qquad \text{on } \Gamma_D$$
 (27)

$$\frac{\partial \tilde{u}}{\partial t} = \Delta \tilde{u} + f \qquad \text{in } \Omega$$

$$\frac{\partial \tilde{u}}{\partial t} = \mu_D(g_D - \tilde{u}) \qquad \text{on } \Gamma_D$$

$$\frac{\partial \tilde{u}}{\partial t} = \mu_N \left(g_N - \frac{\partial \tilde{u}}{\partial n}\right) \qquad \text{on } \Gamma_N$$
(26)

$$\tilde{u} = \tilde{u}_0 \quad \text{in } \Omega, \text{ when } t = 0,$$
 (29)

where μ_D and μ_N are two positive constants. Then we look for the steady state solution. An iterative scheme can therefore be obtained by discretizing the associate time-dependent problem and considering the time just as an iterative parameter.

2.2 Level-set function

In order to keep track of the boundary Γ , we introduce the level set function $\varphi_0 \colon D \to \mathbb{R}$, in such a way:

$$(x,y) \in \stackrel{\circ}{\Omega} \iff \varphi_0(x,y) < 0, \ (x,y) \in \partial\Omega \iff \varphi_0(x,y) = 0.$$

The outward unit normal to the boundary is

$$\mathbf{n} = \frac{\nabla \varphi_0}{|\nabla \varphi_0|}.$$

General references on the level set method for tracking interfaces are, for examples, [31] or [25]. From the level set function, we can obtain the signed distance function φ by fast marching methods [25] or by the reinitialization procedure based on the numerical solution of the following PDE

$$\frac{\partial \varphi}{\partial t} = \operatorname{sgn}(\varphi_0) \left(1 - |\nabla \varphi| \right), \tag{30}$$

as we can see, for instance, in [40, 36, 17]. A signed distance function is preferred to a simple level-set function because sharp gradients are avoided and it is simpler to compute the boundary closest point to a given ghost point. Now we assume that $|\nabla \varphi| = 1$ and suppose we know the signed distance function just at the grid nodes. In practice, Eq. (30) has to be solved for a few time steps, in order to compute the distance function a few grid points away from the boundary. z

2.3 Relaxation operator

Let us introduce some notation. Let $d \in \mathbb{N}$ be the dimension of the problem, $N \geq 1$ be an integer and h = 2/N the spatial step. Let $D_h = \mathbf{j}h, \mathbf{j} = (j_1, \dots, j_d) \in \{-N, N\}^d$ and $\Omega_h = \Omega \cap D_h$ be the discrete versions of D and Ω respectively. D_h is the set of grid points. Two points $\mathbf{x}', \mathbf{x}''$ in D_h are called neighbor if $\sum_{j=1}^{d} \left| x'_j - x''_j \right| = h$. We call ghost point any grid point that is both outside Ω and neighbor to a grid point inside Ω . We call Γ_h the set of all ghost points. Let I_h be a general subset of D_h . We introduce the linear space of grid functions over I_h and we denote it $S(I_h) = \{\mathbf{w}_h \colon I_h \to \mathbb{R}\}$.

From now on, we shall consider d=2, but the results are valid also for d>2.

Then, we write the basic iterative scheme (relaxation scheme) discretizing the time-dependent problem (26)-(29). For any grid point (jh, ih) of Ω_h , we write an equation obtained from the discretization of (26) in such point, using forward Euler in time and central difference in space and taking the maximum time step consented by the CFL condition, i.e. $\Delta t = h^2/4$ (in general it is $\Delta t = h^2/(2d)$):

$$u_{i,j}^{(m+1)} = 1/4 \left(h^2 f_{i,j} + u_{i-1,j}^{(m)} + u_{i+1,j}^{(m)} + u_{i,j-1}^{(m)} + u_{i,j+1}^{(m)} \right).$$
(31)

Eq. (31) is equivalent to discretize directly the first equation of (25) using central difference in space and applying Jacobi iteration scheme.

Since we have used the standard 5-point stencil even for grid point close to the boundary, we have added new unknowns to the linear system (ghost points).

To close the system of equations (31), we must write one equation for each ghost point. This can be done in three simple steps. Let $G \equiv (x_G, y_G)$ be a ghost point.

1. Making use of the signed distance function φ , we can compute the closest boundary point to G, that we call B (see Figure 3), by:

$$B \equiv (x_B, y_B) = G - \mathbf{n}_G \cdot \varphi(G) = G - \left(\frac{\nabla \varphi}{|\nabla \varphi|}\right)\Big|_G \varphi(G), \tag{32}$$

using a second order accurate discretization for $\nabla \varphi$, such as central difference in G.

2. Compute the nine-point stencil (depicted in Fig. 3) in Upwind direction, i.e.:

$$St_G = \left\{ (x_G + s_x k_1 h, y_G + s_y k_2 h) \colon (k_1, k_2) \in \{0, 1, 2\}^2 \right\},\tag{33}$$

where $s_x = \operatorname{sgn}(x_B - x_G)$ and $s_y = \operatorname{sgn}(y_B - y_G)$.

3. Let $\mathcal{L}_{St_G}[u]$ be the biquadratic interpolant of the numerical solution u in the stencil St_G . If $B \in \Gamma_D$, the iteration for the ghost point G will be obtained from the discretization of (27):

$$u_G^{(m+1)} = u_G^{(m)} + \mu_D \Delta t \left(g_D(B) - \mathcal{L}_{St_G}[u^{(m)}](B) \right)$$
(34)

while if $B \in \Gamma_N$, the iteration for the ghost point G will be obtained from the discretization of (28):

$$u_G^{(m+1)} = u_G^{(m)} + \mu_N \Delta t \left(g_N(B) - \left(\nabla \mathcal{L}_{St_G}[u^{(m)}] \cdot \frac{\nabla \mathcal{L}_{St_G}[\varphi]}{|\nabla \mathcal{L}_{St_G}[\varphi]|} \right) \Big|_B \right). \tag{35}$$

The constants μ_D and μ_N are chosen in order to satisfy a CFL condition, i.e. $\mu_D \Delta t < 1$ and $\mu_N \Delta t < 2h/(3\sqrt{2})$ [13]. In numerical tests of Section 3, we choose $\mu_D \Delta t = 0.9$ and $\mu_N \Delta t = 0.9 \cdot 2h/(3\sqrt{2})$. Since $\Delta t = h^2/4$, then $\mu_D = 3.6/h^2$ and $\mu_N = 7.2/(3\sqrt{2}h)$.

Remark 1 (Accuracy of (32)). The accuracy of the evolution of point B in (32) depends on the accuracy at which φ is computed. If φ is known to order h^p , $p \in \{2,3\}$, then B will be computed to the same order of accuracy, provided we are far from singularities in φ .

In Eq. (32) one could omit the term $|\nabla \varphi|$ in the denominator, because $|\nabla \varphi| = 1$ if φ is a signed distance function. However, it is better to keep such term, in case φ is only approximately a signed distance function.

Remark 2 (Upwind stencil). The reason for which we use an Upwind stencil St_G is simple. Let us rewrite the Neumann boundary condition (28) as:

$$\frac{\partial u}{\partial t} + \mu_N \mathbf{n} \cdot \nabla u = \mu_N g_N, \tag{36}$$

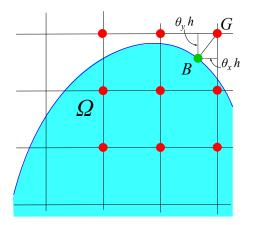


Fig. 3: B is the boundary closest point to G, while the red points are the nine-point stencil in Upwind direction referred to the ghost point G.

in such a way it appears to be an hyperbolic equation which propagates the solution u along the characteristic (normal direction to the boundary) with speed μ_N **n**. Then, it is preferred to use an Upwind stencil in direction $-\mathbf{n}$ to discretize the spatial term, in order to guarantee convergence (see [39] for more detail about Upwind schemes in Conservation laws).

Remark 3 (Reduced stencil). If St_G is not fully contained in $\Omega_h \cup \Gamma_h$, therefore we reduce the nine-point stencil to a smaller stencil, such as a 2×2 stencil or a (less accurate, more robust) three-point stencil. Such a reduction occurs rarely, and does not degrade the whole accuracy of the method (see [13]).

Using the simplified notation, the iterative scheme converges to the solution of the problem:

$$\begin{cases}
-\Delta_h \mathbf{u}_h = \mathbf{f}_h \\
L_h \mathbf{u}_h = \mathbf{g}_h
\end{cases}$$
(37)

where:

- $\mathbf{u}_h \in S(\Omega_h \cup \Gamma_h)$ is the unknown;
- $\Delta_h: S(\Omega_h \cup \Gamma_h) \to S(\Omega_h)$ is the standard discrete version of the Laplacian operator, namely:

$$\Delta_h \mathbf{w}_h(x,y) = \frac{1}{h^2} \left(\mathbf{w}_h(x+h,y) + \mathbf{w}_h(x-h,y) - 4\mathbf{w}_h(x,y) + \mathbf{w}_h(x,y+h) + \mathbf{w}_h(x,y-h) \right)$$

for any $\mathbf{w}_h \in S(\Omega_h \cup \Gamma_h)$ and $(x, y) \in \Omega_h$;

• $\mathbf{f}_h \in S(\Omega_h)$ is defined by $f_h(P) = f(P)$ for any grid point $P \in \Omega_h$;

• $L_h: S(\Omega_h \cup \Gamma_h) \to S(\Gamma_h)$ is the discrete version of boundary conditions, namely:

$$L_{h}\mathbf{w}_{h}(G) = \begin{cases} \mathcal{L}_{St_{G}}[u](B) & \text{if} \quad B \in \Gamma_{D} \\ \left(\nabla \mathcal{L}_{St_{G}}[u] \cdot \frac{\nabla \mathcal{L}_{St_{G}}[\varphi]}{\left|\nabla \mathcal{L}_{St_{G}}[\varphi]\right|}\right) \Big|_{B} & \text{if} \quad B \in \Gamma_{N} \end{cases}$$
(38)

for any $\mathbf{w}_h \in S(\Omega_h \cup \Gamma_h)$ and $G \in \Gamma_h$;

• $\mathbf{g}_h \in S(\Gamma_h)$ is defined by:

$$\mathbf{g}_h(G) = \begin{cases} g_D(B) & \text{if} \quad B \in \Gamma_D \\ g_N(B) & \text{if} \quad B \in \Gamma_N \end{cases}$$

for any ghost point $G \in \Gamma_h$.

2.4 Multigrid approach in 2D

Consider the Poisson problem (25) and suppose we have a discrete approximation of the form (37). Therefore, we are dealing with non-eliminated boundary conditions. Let us introduce, for any spatial step h, an exact solver

$$\mathbf{u}_h = S_h \left(\mathbf{f}_h, \mathbf{g}_h \right) \tag{39}$$

of the system (37), and denote by

$$\Re_h \colon S(\Omega_h \cup \Gamma_h) \times S(\Omega_h) \times S(\Gamma_h) \longrightarrow S(\Omega_h \cup \Gamma_h)$$
 (40)

the relaxation operator, namely the iterative scheme

$$\mathbf{u}_{h}^{(m+1)} = \Re_{h}\left(\mathbf{u}_{h}^{(m)}, \mathbf{f}_{h}, \mathbf{g}_{h}\right) \tag{41}$$

converges to the solution of (37) as $n \to +\infty$. In details, the iteration (41) summarize the iterative scheme (31), (34), (35).

As in one dimensional case, we will intend by (41) the Gauss-Seidel version of (31), (34), (35), in order to deal with a proper *smoother*, and we have to order all points of $\Omega_h \cup \Gamma_h$ in some way. Let us choose the lexicographic ordering (GS-LEX):

$$(x', y') \le (x'', y'') \iff x' < x'' \text{ or } x' = x'', \ y' < y''.$$

The relaxation scheme can be easily extended to more efficient kinds of smoothers, such as Red-Black Gauss-Seidel (see [41]): however, we limit ourselves to study of the GS-LEX smoother.

In order to explain the multigrid approach, we just describe the two-grid correction scheme (TGCS), because all the others schemes, such as V-cycle, W-cycle or Full multigrid, can be easily derived from it (see [41, Sections 2.4, 2.6] for more details). The TGCS consists into the following algorithm:

- 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

$$\mathbf{u}_h := \Re_h \left(\mathbf{u}_h, \mathbf{f}_h, \mathbf{g}_h \right)$$

3. Compute the defect $\mathbf{r}_h \in S(\Omega_h \cup \Gamma_h)$ such as:

$$\mathbf{r}_h = \begin{cases} f_h + \Delta_h \, \mathbf{u}_h & \text{in } \Omega_h \\ g_h - L_h \, \mathbf{u}_h & \text{on } \Gamma_h \end{cases}$$

4. Transfer the defect 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 in the coarser grid

$$\mathbf{e}_{2h} = S_{2h} \left(\mathbf{r}_{2h}^I, \mathbf{r}_{2h}^G \right)$$

where $\mathbf{r}_{2h}^I = \left. \mathbf{r}_{2h} \right|_{\Omega_{2h}}$ and $\mathbf{r}_{2h}^G = \left. \mathbf{r}_{2h} \right|_{\Gamma_{2h}}$;

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

$$\mathbf{u}_h$$
: = $\Re_h (\mathbf{u}_h, \mathbf{f}_h, \mathbf{g}_h)$

We have just to explain the steps concerning grid migration (steps 4 and 6). All the other steps are clear.

2.5 Transfer grid operators

If (40) has the smoothing property, after ν_1 relaxations (step 2 of the algorithm) we have a smooth defect \mathbf{r}_h . Therefore, we can hope to transfer this defect to a coarser grid without losing much information. The defect \mathbf{r}_h defined in the step 3 belongs to $S(\Omega_h \cup \Gamma_h)$. In order to transfer it to a coarser grid, it is convenient to extend in some way this defect in the whole computational domain D_h (i.e. $\mathbf{r}_h \in S(D_h)$), in such a way

we can use the standard full-weighting stencil for the restriction operator $I_{2h}^h: S(D_h) \to S(D_{2h})$, that is (see [41, pag. 42])

$$I_{2h}^{h} = \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}_{2h}^{h} . \tag{42}$$

In general, by the stencil notation

$$I_{2h}^{h} = \begin{bmatrix} \vdots & \vdots & \vdots & \vdots \\ \cdots & t_{-1,-1} & t_{-1,0} & t_{-1,1} & \cdots \\ \cdots & t_{0,-1} & t_{0,0} & t_{0,1} & \cdots \\ \cdots & t_{1,-1} & t_{1,0} & t_{1,1} & \cdots \\ \vdots & \vdots & \vdots & \vdots \end{bmatrix}_{2h}^{h}$$

we will intend the restriction operator I_{2h}^h defined by:

$$I_{2h}^{h}\mathbf{w}_{h}(x,y) = \sum_{(i,j)\in R_{k}} t_{i,j}\mathbf{w}_{h}(x+jh,y+ih),$$

where only a finite number of coefficients $t_{i,j}$ is different from zero, and $R_k \equiv \{-k, \dots, k\}^2$ for some positive integer k. In practice k = 1 allows second order restriction operator.

Let us suppose we have extended the defect to the whole computational domain D_h (as it is carefully described in Sec. 2.6). Anyhow, since we have different operators for inner equations and for boundary conditions, the defect is smooth separately inside Ω_h and along the ghost point Γ_h (or $D_h - \Omega_h$, because of the extension), but it is not smooth in all $\Omega_h \cup \Gamma_h$ (it shows a sharp gradient crossing the boundary, as we can see in Fig. 4).

For this reason, it is convenient to transfer separately on the coarse grid the defect in Ω_h and in $D_h - \Omega_h$. To do that, we introduce a partial grid transfer

$$\tilde{I}_{2h}^h \colon S(D_h) \times \mathcal{P}(D_h) \times S(D_{2h}) \longrightarrow S(D_{2h}),$$

where $\mathcal{P}(D_h)$ is the family of all subset of D_h . Roughly speaking,

$$\mathbf{w}_{2h} = \tilde{I}_{2h}^{h} \left(\mathbf{w}_{h}, I_{h}, \tilde{w}_{2h} \right)$$

means that we transfer \mathbf{w}_h to a coarser grid \mathbf{w}_{2h} using only the points of I_h , leaving unaltered the value in the points out of I_h and already stored in \tilde{w}_{2h} (to better understand, we can think $I_h = \Omega_h$).

In details, let $(x,y) \in D_{2h} \cap I_h$. We focus our attention to the neighborhood of (x,y), that is $\mathcal{N}(x,y) = \{(x+jh,y+ih): j,i=-1,0,1\}$.

Now consider the maximum full rectangle \mathcal{T} with vertices belonging to $\mathcal{N}(x,y)$ and such that $\mathcal{T} \cap D_h \subseteq \mathcal{N}(x,y) \cap I_h$ (see Fig. 5). Therefore, the stencil we use in (x,y) to transfer \mathbf{w}_h to a coarse grid depends on

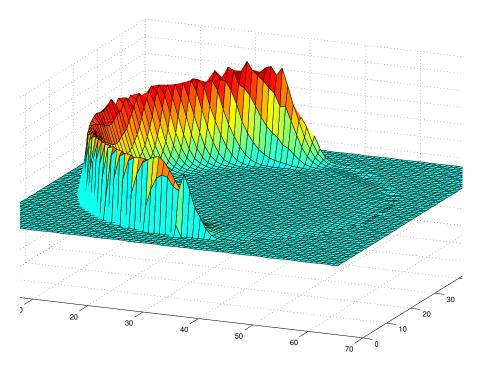


Fig. 4: After few relaxations, the defect show a sharp gradient crossing the boundary (the defect is not yet extended outside), especially for the Neumann boundary. Here the domain is a circle.

the size of \mathcal{T} . In fact, let $\mathcal{T} \cap D_h$ be a 3×3 points (i.e. $\mathcal{N}(x,y) \subseteq I_h$). In this case we can use the standard full-weighting stencil (42).

Now let $\mathcal{T} \cap D_h$ be a 3×2 points. Without loss of generality, we can suppose the vertices of \mathcal{T} are (x+jh,y+ih), with $j \in \{-1,0\}$, $i \in \{-1,1\}$. In this case, the stencil we will use is:

$$\left(\tilde{I}_{2h}^{h}\mathbf{w}_{h}\right)(x,y) = \frac{1}{16} \begin{bmatrix} 2 & 2 & 0\\ 4 & 4 & 0\\ 2 & 2 & 0 \end{bmatrix}_{2h}^{h} (x,y), \tag{43}$$

while, if \mathcal{T} is a 2×2 points, with vertex $(x+jh,y+ih), j,i \in \{-1,0\}$, the stencil will be:

$$\left(\tilde{I}_{2h}^{h}\mathbf{w}_{h}\right)(x,y) = \frac{1}{16} \begin{bmatrix} 0 & 0 & 0 \\ 4 & 4 & 0 \\ 4 & 4 & 0 \end{bmatrix}_{2h}^{h} (x,y), \tag{44}$$

This three case are summarized in Fig. 5 (where $I_h = \Omega_h$). Finally, for all points $(x,y) \in D_{2h} - I_h$, we set $\mathbf{w}_{2h}(x,y) = \tilde{w}_{2h}(x,y)$.

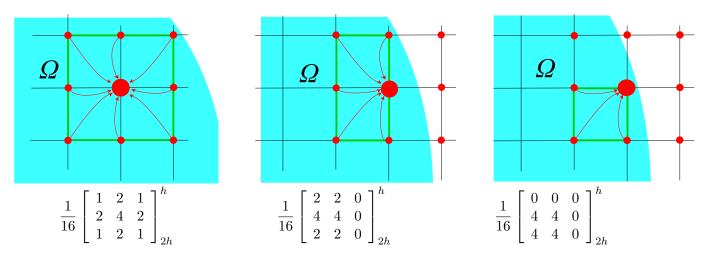


Fig. 5: Upper, the nine points of $\mathcal{N}(x,y)$ and the green boundary of the rectangle \mathcal{T} . The bold red point is on the coarser and finer grids, while the little red points are on the finer grid. The arrows represent the action of the restriction operators. Below, the respective stencil to be used.

In such a way, we can easily define the restriction operator $I_{2h}^h: S(D_h) \to S(D_{2h})$ as follows:

$$I_{2h}^{h}\mathbf{w}_{h}=\tilde{I}_{2h}^{h}\left(\mathbf{w}_{h},D_{h}-\Omega_{h},\tilde{I}_{2h}^{h}\left(\mathbf{w}_{h},\Omega_{h},0\right)\right).$$

Note that the stencils (42), (44), (43) can be derived as tensor products of the one-dimensional restriction (22), (23), (24).

2.6 Extension of the defect

In every ghost point we store the defect of the boundary condition concerning that ghost point. In formulas, we have seen in step 3 of the TGCS algorithm that $\mathbf{r}_h(G) = (\mathbf{g}_h - L_h \mathbf{u}_h)(G)$, for any ghost point G. But $\mathbf{g}_h(G) = g(B)$ and $L_h \mathbf{u}_h(G)$ is the reconstructed boundary condition in B of the boundary operator L (see (38)), where B is the closest boundary point to G (i.e. the orthogonal projection on the boundary). In summary, the defect is stored in a ghost point G, but it is geometrically referred to a boundary point B placed along the normal direction. When we switch to a coarse grid, some ghost point G_1 may not be ghost point in the fine grid, i.e. $\Gamma_{2h} \subseteq \Gamma_h$ is not true (see Fig. 6).

Then, no acceptable value of the defect is stored in G_1 . Indeed, we expect that \mathbf{r}_{2h} has in the ghost point G_1 the defect of the boundary conditions referred to B_1 . Hence, if we extend the defect \mathbf{r}_h outside Ω_h constant along the normal lines to the boundary, we will find $\mathbf{r}_h(G_1)$ as an approximation of the defect of the boundary conditions in B_1 . After coarsening (performed using only points outside Ω_h , as described before), the ghost points of the coarser grid will contain the expected values of the defect.

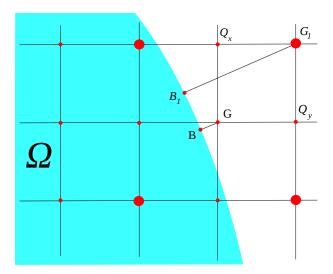


Fig. 6: Red bold and small points are grid points of Ω_h , while red bold points are grid points of Ω_{2h} . G_1 is a ghost point on the coarser grid, but not on the finer grid, then no value of the defect is stored in it. Q_x and Q_y are the two upwind near points to G_1 .

The extension of the defect r_h is performed by solving the transport equation

$$\frac{\partial r}{\partial \tau} + \nabla r \cdot \mathbf{n} = 0$$

in a few steps of a fictitious time τ , where $\mathbf{n} \equiv (n_x, n_y) = \nabla \varphi / |\nabla \varphi|$ is the unit normal vector to the level-set, while r(x,y) is a continuous version of \mathbf{r}_h (i.e. r(x,y) is a continuous function defined in $D-\Omega$ and such that $r(G) = \mathbf{r}_h(G)$ for any ghost point G). In details, we compute few steps of the following iteration scheme:

$$\mathbf{r}_h^{(m+1)}(P) = \mathbf{r}_h^{(m)}(P) + \frac{\Delta \tau}{h} \left(\left(\mathbf{r}_h^{(m)}(P) - \mathbf{r}_h^{(m)}(Q_x) \right) |n_x| + \left(\mathbf{r}_h^{(m)}(P) - \mathbf{r}_h^{(m)}(Q_y) \right) |n_y| \right)$$
(45)

for all $P \in D_h - (\Omega_h \cup \Gamma_h)$, where Q_x and Q_y are the two upwind near points to P, i.e.

$$Q_x = P - \operatorname{sgn}(n_x) h \mathbf{i}, \quad Q_y = P - \operatorname{sgn}(n_y) h \mathbf{j}.$$

However, it is sufficient to perform the iteration (45) only in a narrow band with width 3 h. In order to speed up the extension, we can perform (45) in a Gauss-Seidel fashion, sorting points in $D_h - (\Omega_h \cup \Gamma_h)$ by the distance from the boundary (it can be done using the distance function φ), and starting the computation $\mathbf{r}_h^{(m+1)}(P)$ in (45) from the closest ghost point P to Γ .

2.6.1 Interpolation

Since the interpolation operator acts on the error, which is continuous across the boundary, we just use the standard bilinear interpolation operator:

$$I_h^{2h} = \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \begin{bmatrix} 2h \\ h \end{bmatrix}.$$

3 Numerical tests

In all the following numerical tests we always choose the Dirichlet and Neumann parts of $\Gamma = \partial \Omega$ as:

$$\Gamma_D = \{(x, y) \in \Gamma : x \le 0\}, \quad \Gamma_N = \{(x, y) \in \Gamma : x > 0\}.$$

The Local Fourier Analysis (LFA) is a powerful tool to obtain the theoretically convergence factor by analyzing separately the action of different parts of the multigrid algorithm to high and low frequency components of the error. For a detailed explanation of the LFA, we refer to [41, Ch. 4].

Before to apply the LFA, one has to be sure that the relaxation operator (40) has the *smoothing property*. Roughly speaking, the *smoothing property* is the property to dump high frequency components of the error, in order to make it smooth after few relaxation sweeps.

When the multigrid algorithm applies to a regular rectangular domain, the LFA and smoothing analysis are well studied. In the case of arbitrary domain, as Achi Brandt points out in [41, pag. 587], there are some boundary related difficulties about the discretization and relaxation near the boundary:

- There is no a general smoothing analysis when the boundary is not aligned with the grid;
- The residuals should be reduced near boundaries more than in the interior;
- The coarsest grid has not to be too coarse, because it should catch the curvature of the boundary in order to guarantee the convergence.

Now, we perform a numerical test in order to check if the convergence factor is close to the predicted one by LFA, which is obtained for rectangular domain with periodic boundary conditions, i.e. without taking into account boundary effects. Note that the multigrid algorithm described before may be seen as an iterative scheme:

$$\mathbf{u}_h^{(m+1)} = M_h \, \mathbf{u}_h^{(m)} + \mathbf{b}_h$$

for some matrix M_h and vector \mathbf{b}_h . We call ρ the convergence factor, which is the spectral radius of the matrix M_h . For rectangular domain with periodic boundary conditions and constant coefficients, the convergence factor is said to be local and it is denoted by ρ_{loc} . The convergence factors predicted by LFA for Gauss-Seidel LEX relaxation and FW restriction operator are listed in Table 1 (see [41, pag. 117]).

Table 1: Predicted convergence factor ρ_{loc} by LFA for GS-LEX and FW restriction operator.

$\nu = \nu_1 + \nu_2$	1	2	3	4
$ ho_{loc}$	0.400	0.193	0.119	0.084

In all the numerical tests we perform, the convergence factor is estimated as the ratio of consecutive defects, i.e.:

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

for m very large. In order to avoid difficulties related to numerical instability related to the machine precision, we will always use the homogeneous model problem as a test, namely (25) with $f = g_D = g_N = 0$, and perform the multigrid algorithm starting from an initial guess different from zero. Since we are just interested at the convergence factor and not at the numerical solution itself (which approaches zero indefinitely for homogeneous problem), a reasonable stopping criterion will be

$$\frac{\left|\rho^{(m)} - \rho^{(m-1)}\right|}{\rho^{(m)}} < 10^{-3}.$$

Note that, since we want to study the efficiency of the multigrid components proposed in this paper (smoother, restriction, ...), it is sufficient to study basic kind of multigrid such as V-cycle and W-cycle, while a more efficient algorithm (such as FMG) can be easily derived.

3.1 1D numerical test

Referring to Sec. 1, let us choose $[a,b] = [-0.743,0.843] \subseteq [-1,1]$. The finest grid is obtained dividing the whole computational domain [-1,1] into N=64 subintervals, while the coarsest grid is obtained dividing [-1,1] into Nc=8 subintervals. The computed convergence factors are very close to those ones predicted by LFA (Table 1), namely $\rho=0.185$ for $\nu=2$ and $\rho=0.122$ for $\nu=3$.

3.2 An initial test in 2D

We start testing the multigrid algorithm on a circular domain Ω with center $(\sqrt{2}/20, \sqrt{3}/30)$ and radius r = 0.563 (Fig. 7).

The measured convergence factors for TGCS, V-cycle and W-cycle are listed in Table 2.

As we can see, the measured convergence factor is far from the predicted one by LFA (Table 1). Then, some boundary effect degrades the convergence factor. Note that in 1D such boundary effects do not degrade the convergence factor (Ex. 3.1), because we have only two boundary points, and the degradation is due to the oscillating behavior of the residual on the tangential direction to the boundary, that does not exist in 1D. Then, in 2D we must smooth the error also along the tangential direction to the boundary. To overcome

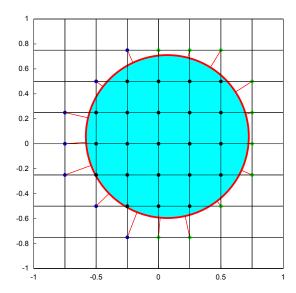


Fig. 7: Circular domain and the coarsest grid used to capture the curvature. Blue ghost points refer to Dirichlet condition, while green ghost points refer to Neumann condition. Red lines are normal to the boundary.

Table 2: Measured convergence factor ρ with $\nu_1 = \nu_2 = 1$ on the left and with $\nu_1 = 2$, $\nu_2 = 1$ on the right.

N	TGCS	V-cycle	W-cycle
64	0.67	0.68	0.71
128	0.68	0.73	0.68
256	0.70	0.71	0.70

N	TGCS	V-cycle	W-cycle
64	0.58	0.72	0.58
128	0.58	0.73	0.59
256	0.61	0.83	0.60

this difficulty, we apply, after a single relaxation and at each grid level, λ extra relaxation sweeps on all ghost points Γ_h and on all inside grid points of Ω_h within $\delta > 0$ distance from the boundary (the extra computational work is O(N), then negligible as $N \to \infty$). It can be proved numerically that a good choice of these parameters will be:

$$\lambda = 5$$
, $\delta = 3 h$.

The explanation of the optimal value $\lambda = 5$ is the following: the degradation observed in Table 2 is an indication that the error decays much slower at the boundary. Assuming that the convergence factor in Table 2 is essentially the convergence factor at the boundary, ρ_B , we want to match it with the convergence factor at the bulk, therefore λ_{opt} is the smallest value of λ for which $\rho_B^{\lambda+1} \leq \rho_I$. The value ρ_I , in turn, can

be computed as the convergence factor for large value of λ .

Investigating the smoothing property, we observe that choosing the initial error as an high frequency component, the error is not smoothed after few relaxation sweeps. While, if we add the extra-relaxations, the error become sufficiently smooth (Figs. 8-9).

3.3 Some numerical results

In this section we confirm numerically the improvement of the convergence factor if we apply extrarelaxations, and we compare the relaxations with other well-knowns alternative such as the Kaczmarz and the block relaxations. In all numerical tests, we choose an arbitrary domain Ω assigning a level-set function φ_0 . Then we reinitialize it by the procedure described in Section 2.2, obtaining the signed distance function φ . Afterwards, we perform the multigrid technique applying the W-cycle algorithm instead of the V-cycle, to ensure the independence of the convergence factor ρ from the step size h (as explained for example in [41, pag. 78]). Several tests are performed for each domain, based on the different size of the finest and coarsest grids. The finest grid is obtained dividing the whole computational domain D into N subintervals in each Cartesian direction, while the coarsest grid is obtained replacing N with N_c . The solution on the coarsest grid is obtained by a direct solver.

3.3.1 Circular domain

In this case we can choose as a level-set function directly the signed distance function, which is known analytically:

$$\varphi(x,y) = \sqrt{(x - \sqrt{2}/20)^2 + (y - \sqrt{3}/30)^2} - 0.563.$$

The zero level-set is represented in Fig 10 (top-left). Different value of the convergence factor are listed in Table 3 (for $\nu = \nu_1 + \nu_2 = 2$ and $\nu = 3$). They are really improved with respect to those obtained without extra-relaxations (Table 2) for the same test.

Table 3: Convergence factor of the numerical test of Section 3.3.1. We use $N \times N$ number of grid points in the finest grid; $N_c \times N_c$ number of grid points in the coarsest grid. Left table: $\nu = \nu_1 + \nu_2 = 2$, right table $\nu = \nu_1 + \nu_2 = 3$.

	N	16	32	64	128	256
N_c						
8		0.052	0.053	0.11	0.13	0.14
16			0.061	0.11	0.13	0.14
32				0.11	0.13	0.14
64					0.13	0.14
128						0.14

	N	16	32	64	128	256
N_c						
8		0.06	0.03	0.09	0.08	0.08
16			0.04	0.09	0.08	0.08
32				0.09	0.08	0.08
64					0.09	0.08
128						0.09

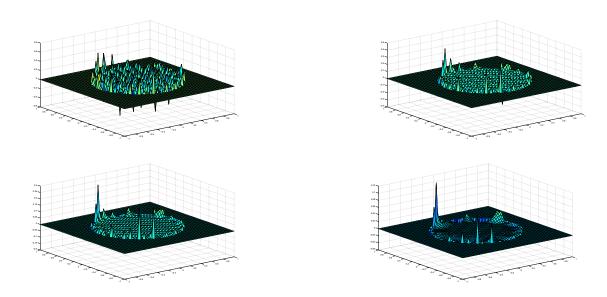


Fig. 8: High frequency initial error after 1 (up-left), 3 (up-right), 5 (down-left), 10 (down-right) relaxation sweeps and $\lambda=0$ extra-relaxations.

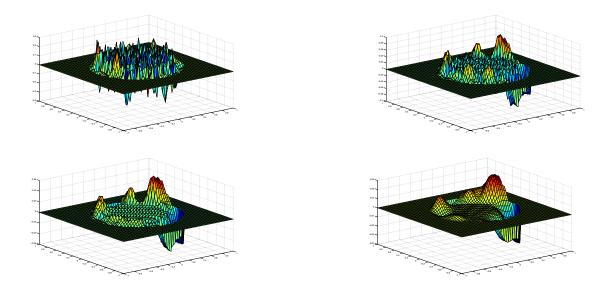


Fig. 9: High frequency initial error after 1 (up-left), 3 (up-right), 5 (down-left), 10 (down-right) relaxation sweeps and $\lambda = 5$ extra-relaxations.

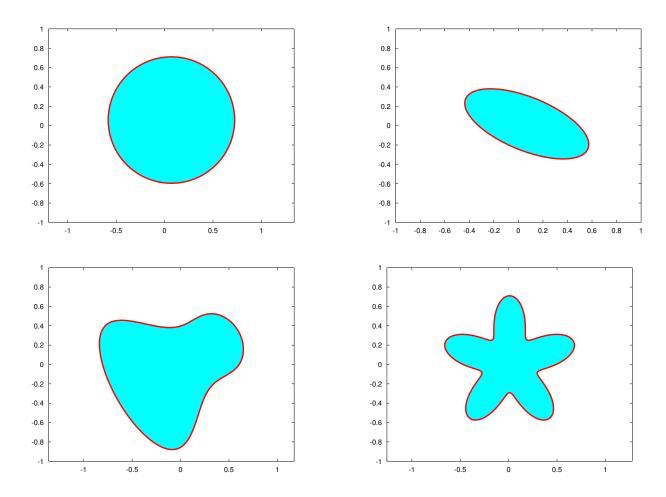


Fig. 10: Different domains used in the numerical tests: Example 3.3.1 (top-left), 3.3.3 (top-right), 3.3.4 (bottom-left), 3.3.5 (bottom-right).

3.3.2 Comparison with the Kaczmarz and the block relaxations

Note that the relaxation scheme (7), (8), (9) is composed by a Gauss-Seidel iteration over inner grid points and a suitable relaxation over ghost points (boundary conditions). As an alternative to the relaxation of the boundary condition, we can use the Kaczmarz relaxation [26] near the boundary, which is known to be unconditionally convergent. Let us recall the Kaczmarz iteration scheme for a subset of equations

 $\mathcal{J} \subseteq \{\ldots, N_i + N_q\}$ of a linear system Lu = f:

$$u^{TEMP} = u^{(m)},$$
for $j \in \mathcal{J}$ do: $u^{TEMP} := u^{TEMP} + \frac{f_j - \langle l_j, u^{TEMP} \rangle}{\|l_j\|_2^2} l_j^T,$

$$u^{(m+1)} = u^{TEMP}.$$

The symbol $\langle \cdot, \cdot \rangle$ denotes the inner product operator and l_j is the j-th row of the matrix L. If we choose $J = \{ \dots, N_i + N_g \}$ then we obtain the classical Kaczmarz relaxation scheme for the solution of the linear system Lu = f, and the iteration scheme is equivalent to a Gauss-Seidel relaxation for the system $L^T Lu = L^T f$. In our case, one iteration of the alternative relaxation we want to study is composed as follows: we perform a Gauss-Seidel sweep in the interior of the domain, followed by λ Kaczmarz iterations over ghost points and inner points close to the boundary (say within δ distance from the boundary).

Another alternative is represented by the block relaxation [16]. As we point out in [13], the elimination of the boundary conditions is hard to perform in high dimensions, while in one dimension it is a trivial task and leads to a diagonally dominant linear system. A middle ground between the elimination of the boundary conditions and the relaxation operator we use in this paper is the block relaxation. Let us describe it in details. For each grid point $P \in \Omega_h \cup \Gamma_h$ we choose a stencil $St_P \subseteq \Omega_h \cup \Gamma_h$. For instance, if $P \in \Omega_h$ we choose $St_P = St_{P9} \cap (\Omega_h \cup \Gamma_h)$, where St_{P9} is the 3×3 stencil centered at P, else if $P \in \Gamma_h$ we choose the stencil St_P defined in (33). One iteration of the alternative relaxation is composed as follows. We perform a Gauss-Seidel sweep in the interior of the domain except in grid points within δ distance from the boundary. For each grid point $P \in \Omega_h \cup \Gamma_h$ within δ distance from the boundary we rewrite the linear system Lu = f as follows (by a permutation of rows):

$$Lu = f \Leftrightarrow \left(\begin{array}{cc} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{array}\right) \left(\begin{array}{c} u_1 \\ u_2 \end{array}\right) = \left(\begin{array}{c} f_1 \\ f_2 \end{array}\right)$$

where u_1 is referred to those grid points belonging to St_P . Therefore, we update the values of u_1 as:

$$u_1 = A_{1,1}^{-1} (f_1 - A_{1,2} u_2).$$

We perform a comparison between the relaxation proposed in this paper (that we call *new iteration* in the following plots) and the two alternative relaxation described above. Such a comparison is carried out in terms of smoothing factor and convergence factor. We perform the comparison using the TGCS for the test case of the circular domain 3.3.1 with N = 64.

In Fig. 11 we plot the smoothing factor μ for the three iteration schemes, which is estimated by the ratio of subsequent defects after each iteration, i.e.

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

In practice, we perform only the iteration schemes, without taking into account the effects of the multigrid procedure. In order to better capture the behavior of the smoothing factor, we choose an initial guess being highly oscillant, for example $u = \sin(40\pi x)\sin(50\pi y)$.

In Fig. 12 we depict the convergence factor ρ for the Kaczmarz and the new iteration against the number of extra-relaxations λ (for comparison, we also plot the convergence factor of the block relaxation as an horizontal line, since it does not depend on λ). After five extra-relaxations, the new iteration reaches a plate configuration, since it achieves the convergence factor of the Gauss-Seidel smoother for inner equations, i.e. the convergence factor predicted by the LFA (see Table 1). The Kaczmarz iteration falls down slower, while the block iteration already provides the optimal convergence factor. The computational cost of five point-iterations of the new method is considerably lower than the cost of one block-iteration.

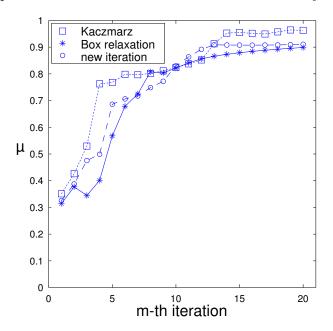


Fig. 11: Smoothing factor μ against the number of iterations for the three iterative schemes: Kaczmarz relaxation, Block relaxation, new iteration.

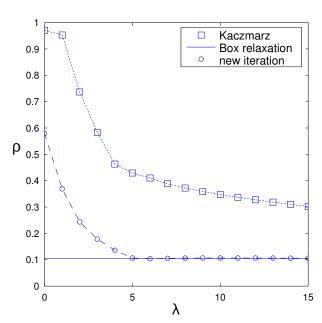


Fig. 12: Convergence factor ρ of the entire multigrid against the number of extra-relaxations λ for the three iterative schemes: Kaczmarz relaxation, Block relaxation, new iteration.

3.3.3 Ellipsoidal domain

The level-set function is:

$$\varphi(x,y) = \frac{(X(x,y) - \sqrt{2}/20)^2}{0.563^2} + \frac{(Y(x,y) - \sqrt{3}/30)^2}{0.263^2} - 1$$

where

$$X(x,y) = \cos(\pi/6) x - \sin(\pi/6) y$$
, $X(x,y) = \sin(\pi/6) x + \cos(\pi/6) y$,

and the zero level-set is represented in Fig. 10 (top-right). The convergence factor obtained are listed in Table 4 (for $\nu = \nu_1 + \nu_2 = 2$ and $\nu = 3$). We observe as the convergence factor degrade choosing a coarsest grid too much coarse, but starting from a certain level of coarsest grid it is relatively close to the predicted convergence factor by LFA (Table 1).

Table 4: Convergence factor of the numerical test of Section 3.3.3. We use $N \times N$ number of grid points in the finest grid; $N_c \times N_c$ number of grid points in the coarsest grid. Left table: $\nu = \nu_1 + \nu_2 = 2$, right table $\nu = \nu_1 + \nu_2 = 3$.

	N	16	32	64	128	256
N_c						
8		0.34	0.09	0.14	0.14	0.15
16			0.65	0.45	0.19	0.15
32				0.14	0.14	0.15
64					0.15	0.15
128						0.15

	N	16	32	64	128	256
N_c						
8		0.44	0.06	0.12	0.11	0.09
16			0.55	0.30	0.09	0.09
32				0.13	0.10	0.09
64					0.12	0.08
128						0.09

3.3.4 Saddle-shaped domain

The level-set function is:

$$\varphi(x,y) = 9\left(\frac{1}{2}x - \frac{\sqrt{3}}{2}y\right)^2 + \left(\frac{3\sqrt{3}}{2}x + \frac{3}{2}y - 1\right)^2 \sin\left(\frac{3\sqrt{3}}{2}x + \frac{3}{2}y - 1\right) - 1$$

and the zero level-set is represented in Fig. 10 (bottom-left). The convergence factor obtained for $\nu = \nu_1 + \nu_2 = 3$ are listed in Table 5 (left). Also in this case, only if we choose N=16 and $N_c=8$ (which actually is TGCS) the convergence factor is degraded.

3.3.5 Flower-shaped domain

The level-set function is:

$$\varphi = r - 0.5 - \frac{y^5 + 5x^4y - 10x^2y^3}{5r^5}, \quad r = \sqrt{x^2 + y^2}$$

and the zero level-set is represented in Fig. 10 (bottom-right). The convergence factor obtained for $\nu = \nu_1 + \nu_2 = 3$ are listed in Table 5 (right). This is the hardest numerical test, because of the indentation of the boundary. We need to start from a coarsest level $N_c = 32$ to correctly capture the boundary profile and to make the discretization accurate.

Table 5: Convergence factor of the numerical test of Sections 3.3.4 (left) and 3.3.5 (right). We use $N \times N$ number of grid points in the finest grid; $N_c \times N_c$ number of grid points in the coarsest grid. In this test we use $\nu = \nu_1 + \nu_2 = 3$. For the flower-shaped domain, if $N_c = 8$ the multigrid does not converge.

	N	16	32	64	128	256
N_c						
8		0.36	0.08	0.09	0.12	0.09
16			0.12	0.09	0.12	0.09
32				0.09	0.12	0.09
64					0.13	0.09
128						0.09

	N	16	32	64	128	256
N_c						
8		n.c.	n.c.	n.c.	n.c.	n.c.
16			0.89	0.75	0.50	0.25
32				0.49	0.25	0.12
64					0.24	0.11
128						0.09

Conclusion

A multigrid technique for Poisson equation on an arbitrary domain and mixed boundary conditions is presented. This multigrid strategy can be applied to a general framework of ghost-point method in a regular Cartesian grid, in case of non-eliminated boundary conditions. Suitable transfer operators for inside equations and boundary conditions are provided. The convergence rate is improved by adding some extra-relaxations on the ghost points and in a narrow band of inside grid points close to the boundary. Numerical tests on different geometries have been performed. On simple domains (such as circle or ellipse) the optimal convergence factor is reached even if the method is used on very coarse grids, while for more complex domains (such as the flower-shaped one) the optimal convergence factor is obtained only on sufficiently fine grids. A comparison with other treatments of the boundary condition smoothing procedure has been carried out (Kaczmarz and Block relaxation), confirming that the smoother proposed in this paper is better in terms of convergence factor, and not worse in terms of smoothing factor.

The application of this method to solve the pressure equation coming from the projection method of Chorin [10, 11] in the framework of the incompressible Navier-Stokes equation is in preparation. Several extensions of the discretization technique and multigrid approach are presently under investigation. We mention the case of discontinuous coefficients, which models, for example, a system composed by different materials separated by an interface; in such a case the method is suitably modified in order to achieve second order accuracy and a convergence factor being independent on the jump in the coefficient. A preliminary result can be found in [14]. Another extension concerns the convection-diffusion equation in a moving domain, in order to study applications modeled by a Stefan-type problem. All these extensions will be coupled with the use of Adaptive Mesh Refinement to obtain accurate solution in the case of domain with complex boundary.

References

- [1] L. Adams and Z. Li. The immersed interface/multigrid methods for interface problems. *Journal of Scientific Computing*, 24:463–479, 2002.
- [2] 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.
- [3] A. Brandt. Rigorous Quantitative Analysis of Multigrid, I: Constant Coefficients Two-Level Cycle with L2-Norm. SIAM Journal on Numerical Analysis, 31:1695–1730, 1994.
- [4] W. L. Briggs, V. E. Henson, and S. F. McCormick. A Multigrid Tutorial. SIAM, 2000.
- [5] R. E. Caffisch, M. F. Gyure, B. Merriman, S. J. Osher, C. Ratsch, D. D. Vvedensky, and J. J. Zinck. Island dynamics and the level set method for epitaxial growth. *Applied Mathematics Letters*, 4:13–22, 1999.
- [6] L. A. Catalano, A. Dadone, V. S. E. Daloiso, and D. Scardigno. A multigrid procedure for Cartesian ghost-cell methods. *International Journal for Numerical Methods in Fluids*, 58:743–750, 2008.
- [7] 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.
- [8] 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.
- [9] Y. Chiang, B. V. Leer, and K. G. Powell. Simulation of unsteady inviscid flow on an adaptively refined Cartesian grid. In *AIAA Paper*, 1999.
- [10] A. Chorin. Numerical solution of the NavierStokes Equations. *Mathematics of Computation*, 22:745–762, 1968.
- [11] A. Chorin. A numerical method for solving incompressible viscous flow problems. *Journal of Computational Physics*, 135:115–125, 1997.
- [12] D. Clarke, M. Salas, and H. Hassan. Euler calculations for multielement airfoils using Cartesian grids. AIAA Journal, 24:353–358, 1986.
- [13] 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.

- [14] A. Coco and G. Russo. Second order multigrid methods for elliptic problems with discontinuous coefficients on an arbitrary interface, I: one dimensional problems. *Numerical Mathematics: Theory, Methods and Applications*. Accepted.
- [15] R. Courant, K. Friedrichs, and H. Lewy. On the partial difference equations of mathematical physics. *IBM J. Res. Develop.*, 11:215–234, 1967.
- [16] J. Dendy. Black box multigrid. Journal of Computational Physics, 48:366–386, 1982.
- [17] A. du Chéné, C. Min, and F. Gibou. Second-Order Accurate Computation of Curvatures in a Level Set Framework Using Novel High Order Reinitialization Schemes. *Journal of Scientific Computing archive*, 35:114–131, 2008.
- [18] 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.
- [19] 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.
- [20] 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.
- [21] W. Hackbusch. Multi-grid methods and applications. Springer, 1985.
- [22] W. Hackbusch. Elliptic Differential Equations: Theory and Numerical Treatment. Springer, 2003.
- [23] C. Helzel, M. J. Berger, and L. R. J. A high-resolution rotated grid method for conservation laws with embedded geometries. (English summary). SIAM J. Sci. Comput., 26:785–809, 2005.
- [24] 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.
- [25] J.Sethian. Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geometry, Fluid Mechanics, Computer Vision and Materials Science. Cambridge University Press, 1999.
- [26] S. Kaczmarz. Angenherte Auflsung von Systemen linearer Gleichungen. Bulletin International de l'Acadmie Polonaise des Sciences et des Lettres. Classe des Sciences Mathmatiques et Naturelles. Srie A, Sciences Mathmatiques, 35:355–357, 1937.
- [27] 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.

- [28] R. Löhner, J. R. Cebral, F. E. Camelli, S. Appanaboyina, J. D. Baum, E. L. Mestreau, and O. A. Soto. Adaptive embedded and immersed unstructured grid techniques. Comput. Methods Appl. Mech. Engrg., 197:217–2197, 2008.
- [29] Z. H. Ma, L. Qian, D. M. Causon, H. B. Gu, and C. G. Mingham. A Cartesian ghost-cell multigrid Poisson solver for incompressible flows. *International Journal for Numerical Methods in Engineering*, 85:230–246, 2011.
- [30] R. H. Nochetto, M. Paolini, and C. Verdi. An adaptive finite element method for two-phase stefan problem in two space dimensions. part ii: Implementation and numerical experiments. SIAM J. Sci. Stat. Comput., 12:1207–1244, 1991.
- [31] S. Osher and R. Fedkiw. Level Set Methods and Dynamic Implicit Surfaces. Springer-Verlag New York, Applied Mathematical Sciences, 2002.
- [32] 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.
- [33] C. S. Peskin. Numerical analysis of blood flow in the heart. *Journal of Computational Physics*, 25:220–252, 1977.
- [34] A. Quarteroni. Numerical models for differential problems. Springer, 2009.
- [35] A. Quarteroni and R. Sacco. Numerical approximation of partial differential equations. Springer, 1994.
- [36] G. Russo and P. Smereka. A remark on computing distance functions. *Journal of Computational Physics*, 163:51–67, 2000.
- [37] A. Schmidt. Computation of three dimensional dendrites with finite elements. *Journal of Computational Physics*, 125:293–312, 1996.
- [38] G. H. Shortley and R. Weller. The numerical solution of laplace's equation. *J. Appl. Phys.*, 9:334–348, 1938.
- [39] J. C. Strikwerda. Finite Difference Schemes and Partial Difference Equations. Second Edition. SIAM, 2004.
- [40] M. Sussman, P. Smereka, and S. Osher. A level set approach for computing solutions to incompressible 2-phase flow. *Journal of Computational Physics*, 114:146–159, 1994.
- [41] U.Trottemberg, C. Oosterlee, and A. Schuller. Multigrid. Academic Press, 2000.
- [42] A. Wiegmann, Z. Li, and R. LeVeque. Crack Jump Conditions for Elliptic Problems. Applied Mathematics Letters, 12:81–88, 1999.