J Sci Comput (2013) 55:1–15
DOI 10.1007/s10915-012-9619-2

A Multigrid Method on Non-Graded Adaptive Octree and Quadtree Cartesian Grids

Maxime Theillard · Chris H. Rycroft · Frédéric Gibou

Received: 10 January 2011 / Revised: 6 December 2011 / Accepted: 8 June 2012 /
Published online: 26 June 2012

Abstract In order to develop efficient numerical methods for solving elliptic and parabolic problems where Dirichlet boundary conditions are imposed on irregular domains, Chen et al. (J. Sci. Comput. 31(1):19–60, 2007) presented a methodology that produces second-order accurate solutions with second-order gradients on non-graded quadtree and octree data structures. These data structures significantly reduce the number of computational nodes while still allowing for the resolution of small length scales. In this paper, we present a multigrid solver for this framework and present numerical results in two and three spatial dimensions that demonstrate that the computational time scales linearly with the number of nodes, producing a very efficient solver for elliptic and parabolic problems with multiple length scales.

Keywords Multigrid method · Poisson's equation · Non-graded adaptive grid · Octrees · Quadtrees · Second-order discretization · Complex geometry

1 Introduction

Many problems in science and engineering require the solution of elliptic and parabolic equations on, possibly moving, irregular domains. Examples include the solution of the

M. Theillard (✉) · F. Gibou
Department of Mechanical Engineering, University of California, Santa Barbara, CA 93106, USA
e-mail: maxime_theillard@umail.ucsb.edu

F. Gibou
e-mail: fgibou@engineering.ucsb.edu

C.H. Rycroft
Department of Mathematics, University of California and LBNL, Berkeley, CA 94720, USA
e-mail: chr@math.berkeley.edu

C.H. Rycroft
Department of Mathematics, Lawrence Berkeley Laboratory, Berkeley, CA 94720, USA

F. Gibou
Department of Computer Science, University of California, Santa Barbara, CA 93106, USA

Poisson equation in the context of free surface flow, the simulation of the Stefan problem with application to crystal growth, or the solution of the Poisson–Boltzmann equation with application to electrostatics around large molecules in an ionic solution. More often than not, these problems involve different length scales that must be resolved. These can originate from the boundary conditions imposed on the irregular domain's boundary, which in turn reduce the otherwise smooth nature of solutions; or the rapid variation of the solution near the boundary, as is the case for the electric double layer surrounding particles, molecules or the walls of nano-channels. From the computational point of view, it has long since been recognized that uniform grids are not desirable when different length scales are present, since imposing a high resolution uniformly in the domain may lead to computationally intractable systems.

In such cases, adaptive mesh refinement (AMR) is a more judicious choice. AMR was pioneered by Berger and Oliger [2]. Originally developed for compressible flows, a block-structured AMR approach discretizes the computational domain uniformly with a coarse grid, and then refines rectangular blocks of uniform grids with finer resolutions where needed.

Grids based on quadtree and octree data structures do not impose patches of uniform grids but rather allow the grid cells to continuously change in size, producing meshes with significantly fewer computational nodes than in block AMR frameworks. In some studies graded grids have been employed [3], where the size ratio between adjacent cells is not allowed to be greater than two, which can simplify the discretizations of different operators. Recently, there has been a thrust in developing numerical methods for non-graded grids, where the size ratio between adjacent cells is not constrained [1, 4–7]. Non-graded grids can be generated efficiently and straightforwardly, are more versatile and usually generate fewer grid points than their graded counterparts, which can save computational resources.

Multigrid methods, first introduced by Brandt [8, 9] provide a framework that is particularly efficient for solving linear systems. The multigrid method is an iterative approach in which a hierarchy of progressively coarser grids is introduced. These grids can be employed to damp out errors over a wide spectrum of frequencies, allowing for extremely fast convergence, with a theoretical complexity that scales linearly with number of grid points. Because they employ grids of multiple resolution, they are particularly well-suited to adaptive settings, and this concept was considered in Brandt's original work. Recently, several studies have investigated multigrid methods on octree grids [10, 11], although these have focused on cell-based discretizations on graded grids. In this paper, we present the results of an implementation of a multigrid method on node-based non-graded quadtree and octree grids. A Dirichlet boundary condition is imposed on the boundary of the irregular domain following the work of Chen [1]. We present numerical results in two and three spatial dimensions.

2 Discretization on Quadtree and Octree Data Structures

2.1 Domain Description and Grid Generation

We represent the boundary of the domain by making use of a level set function ϕ [12–14]. Consider a computational domain Ω as the union of two subdomains Ω^+ and Ω^- separated by an interface Γ, and define Ω^- as the set of points $\mathbf{x}$ where $\phi(\mathbf{x}) < 0$, Ω^+ as the set of points $\mathbf{x}$ where $\phi(\mathbf{x}) > 0$ and Γ as the set of points $\mathbf{x}$ where $\phi(\mathbf{x}) = 0$. Here, we will consider that the irregular domain is defined as $\Omega^- \cup \Gamma$.

To generate the grid, a refinement criterion must be chosen to determine when to split a cell. Although several refinement criteria could be defined, we are principally interested in automatically refining the mesh where the interface lies. This is particularly important in problems where the solution near the interface varies rapidly. Based on the work of Strain [15] and Min [16], Min and Gibou [7] proposed the following simple refinement criteria for generating such grids: Split any cell C if

$$\min_{v\in\text{vertices}(C)} \left|\phi(v)\right| \leq \text{Lip}(\phi)\cdot\text{diag-size}(C), \tag{1}$$

where diag-size(C) refers to the length of the diagonal of the current cell C, v refers to a vertex (node) of the current cell, and Lip(ϕ) is the Lipschitz constant of ϕ. This process is depicted in Fig. 1. We also note that it is straightforward to generate a grid with a uniform band around the interface. For example, to build a uniform grid with half-band width $a\Delta x$, one can successively apply the criterion (1) to a temporary ϕ translated by $\pm\Delta x, \pm 2\Delta x, \ldots, \pm a\Delta x$.

Quadtree and octree data structures are efficient in storing adaptive grids in two and three spatial dimensions, respectively [17]. The root is initially associated to the entire computational domain and each cell is then split recursively into four (2D) or eight (3D) identical cells, called the children of the cell. The level of each cell is defined as the number of splittings necessary to create it. A grid for which the difference of level between two adjacent cells is less than or equal to one is referred to as graded, and non-graded otherwise. In this work we consider the most general non-graded grids.

2.2 Discretization of Standard Operators

The main difficulty in discretizing general differential operators on non-graded grids is due to nodes where at least one of the neighbors in the Cartesian directions is missing; these nodes are referred to as T-junctions or hanging nodes. Figure 2 represents the most general configuration of neighboring nodes in the case of an octree, where the T-junction node v_0 has four regular neighboring nodes and two ghost nodes. Min and Gibou [7] derived a third-order accurate interpolation of a node-sampled function $f : \{v_i\} \to \mathbb{R}$ at the ghost nodes v_4 and v_5 as

$$f_4^G = \frac{s_7 f_8 + s_8 f_7}{s_7 + s_8} - \frac{s_7 s_8}{s_3 + s_6}\left(\frac{f_3 - f_0}{s_3} + \frac{f_6 - f_0}{s_6}\right), \tag{2}$$

$$f_5^G = \frac{s_{11}s_{12}f_{11} + s_{11}s_9 f_{12} + s_{10}s_{12}f_9 + s_{10}s_9 f_{10}}{(s_{10}+s_{11})(s_9+s_{12})} - \frac{s_{10}s_{11}}{s_3+s_6}\left(\frac{f_3-f_0}{s_3}+\frac{f_6-f_0}{s_6}\right) - \frac{s_9 s_{12}}{s_1+s_4}\left(\frac{f_1-f_0}{s_1}+\frac{f_4^G-f_0}{s_4}\right), \tag{3}$$

where s_i refers to the distance between v_0 and v_i. The definition of the ghost nodes allows the definition of central finite differences for f_x, f_y, f_z, f_{xx}, f_{yy} and f_{zz} at every node. By referring to Fig. 2, the second-order derivatives can be discretized as

$$D_{xx} f(v_0) = \left(\frac{f_1 - f_0}{s_1}\right)\left(\frac{2}{s_1+s_4}\right) - \left(\frac{f_0 - f_4^*}{s_4}\right)\left(\frac{2}{s_4+s_1}\right), \tag{4}$$

$$D_{yy} f(v_0) = \left(\frac{f_2 - f_0}{s_2}\right)\left(\frac{2}{s_2+s_5}\right) - \left(\frac{f_0 - f_5^*}{s_5}\right)\left(\frac{2}{s_5+s_2}\right), \tag{5}$$

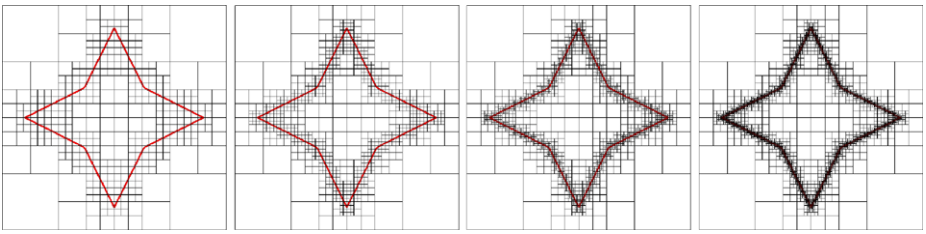


Fig. 1 The refinement process for a fixed interface. Starting from a given grid (*left*), we recursively refine the grid using the refinement criterion given in Eq. (1)

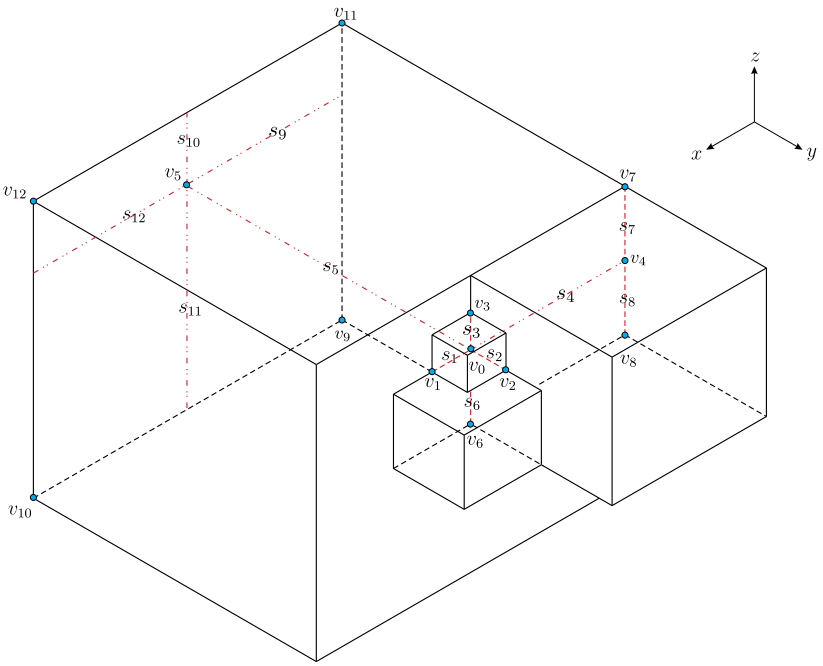


Fig. 2 The most general octree neighborhood configuration around a grid node v_0. In this example, v_1, v_2, v_3, and v_6 are nodes of the grid, v_4 is a ghost node lying on an edge, while v_5 is a ghost node lying on a face

$$D_{zz} f(v_0) = \left(\frac{f_3 - f_0}{s_3}\right)\left(\frac{2}{s_3 + s_6}\right) - \left(\frac{f_0 - f_6}{s_6}\right)\left(\frac{2}{s_6 + s_3}\right), \tag{6}$$

where f_k^* refers to either the actual node value or the ghost node value depending on availability. The Poisson operator can be constructed as the sum of these three terms.

2.3 Discretization Near the Interface

Consider the case depicted in Fig. 3, where the node v_3 is outside the irregular domain Ω^-. By enforcing a band of uniform cells along the interface, we guarantee that v_0 has a direct neighboring node in each direction. The discretization is modified to directly include the

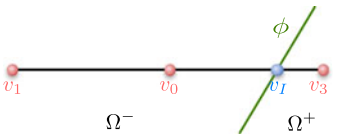


Fig. 3 Schematic representing the case where the interface crosses between the nodes v_0 and v_3 at location v_I

Dirichlet condition at the interface point v_I. For example, we define

$$f_{xx}(v_0) = \left(\frac{f_1 - f_0}{s_1} + \frac{f_I - f_0}{s_I} \right) \frac{2}{s_I + s_1}, \tag{7}$$

where the distance s_I between v_0 and v_I is given by

$$s_I = \begin{cases} \frac{-\phi_x + \sqrt{\phi_x^2 - 2\phi_{xx}\phi_0}}{\phi_{xx}} & \text{if } |\phi_{xx}| > \epsilon \quad (\text{for } \epsilon \text{ small}) \\ \frac{-\phi_0}{\phi_x} & \text{otherwise.} \end{cases}$$

2.4 Properties of the Discretized Poisson Operator

The discretizations introduced above can be shown to lead to weakly diagonally dominant linear systems for the Poisson operator, which will be useful in the subsequent convergence analysis. A matrix A with entries A_{ij} is said to be weakly diagonally dominant if for all i,

$$|A_{ii}| \geq \sum_{i \neq j} |A_{ij}| \tag{8}$$

with the equality being strict for at least one value of i.

To establish weak diagonal dominance, consider a given grid point and the corresponding row in the matrix of the Poisson operator, which is the sum of the three second-order derivative operators shown in Eqs. (4) to (6). Consider first the case when no ghost values are needed. Each of the three equations has two negative contributions to the diagonal term associated with f_0. In addition, each equation gives two positive terms for off-diagonal terms associated with values of f_k for $k \neq 0$, whose sum will be equal and opposite the corresponding diagonal term. Hence equality will be achieved in Eq. (8).

Showing that the weak diagonal dominance property holds for nodes where ghost values are needed can be carried out by examining how the terms in Eqs. (2) and (3) alter the balance of Eq. (8). Both of the interpolation formulae at the ghost nodes involve constructing a ghost value as a linear interpolation of nearby values, plus several terms of the form $(f_k - f_0)$ for some $k \neq 0$; these add a positive contribution to the diagonal entry plus an equal and opposite negative contribution to the off-diagonal terms. The expression for f_4^G has terms which will that will modify the terms associated with D_{zz}, while the expression for f_5^G has terms associated with D_{xx} and D_{zz}. In order for weak diagonal dominance to hold, it is necessary that these ghost value contributions are smaller in magnitude than the regular contributions, so that the each off-diagonal matrix entry remains positive. If any off-diagonal entry becomes negative, then there will be a corresponding positive contribution to the diagonal entry, and Eq. (8) will be violated.

First, consider the case when node v_4 is missing but v_5 is present. Eq. (2) can be rewritten as

$$f_4^G = \frac{s_7 s_8}{s_3 s_6} f_0 + \frac{s_7}{s_7 + s_8} f_8 + \frac{s_8}{s_7 + s_8} f_7 - \frac{s_7 s_8}{(s_3 + s_6) s_3} f_3 - \frac{s_7 s_8}{(s_3 + s_6) s_6} f_6.$$

The terms involving f_7 and f_8 sum to one, and thus they will contribute the same amount to $\sum_{i\neq j}|A_{ij}|$ as if f_4 was present. The contribution from the remaining terms to the Poisson operator will be

$$\frac{2s_7s_8}{(s_1+s_4)s_4}\left(\frac{1}{s_3s_6}f_0-\frac{1}{(s_3+s_6)s_3}f_3-\frac{1}{(s_3+s_6)s_6}f_6\right) \tag{9}$$

and it can be seen that the coefficient of f_0 is equal and opposite the sum of the coefficients of f_3 and f_6. The terms from Eq. (6) can be written as

$$-\frac{2f_0}{s_3s_6}+\frac{2}{(s_3+s_6)s_3}f_3+\frac{2}{(s_3+s_6)s_6}f_6. \tag{10}$$

From Fig. (2) it can be seen that $s_4 > s_7$ and $s_4 > s_8$. Hence it follows that when the contributions from Eqs. (9) and (10) are added to the Poisson operator, the terms for f_3 and f_6 will remain positive while the terms for f_0 will remain negative. Thus the modifications to this row of the linear system will still satisfy the weak diagonal dominance condition.

If v_5 is missing but v_4 is present, the same argument can be applied. The terms involving f_9, f_{10}, f_{11}, and f_{12} in Eq. (3) sum to one, and have the same contribution as f_5 would if it were present. The remaining two bracketed terms will modify the size of terms in the discretizations of D_{xx} and D_{zz} but since s_5 is larger than s_9, s_{10}, s_{11}, s_{12} the same argument can be applied to deduce that the modifications will not be large enough to switch the sign of the contributions.

The most complex case occurs when ghost values are needed for both v_4 and v_5. To begin, consider the contribution from f_5^G. The second bracketed term has the form

$$\frac{2s_9s_{12}}{(s_2+s_5)s_5}\left(\frac{1}{s_1s_4}f_0-\frac{1}{(s_1+s_4)s_1}f_1-\frac{1}{(s_1+s_4)s_4}f_4^G\right).$$

Overall, this bracketed term will modify the discretization of D_{xx}, but in the same manner as considered above, this is small enough as to not switch the signs of the matrix terms. Coupled with the f_4^G term in the discretization of D_{xx}, this leads to a total contribution of f_4^G as

$$\frac{2}{(s_1+s_4)s_4}\left(1-\frac{s_9s_{12}}{(s_2+s_5)s_5}\right).$$

Both the bracketed term in the definition of f_4^G and the first bracketed term in the definition of f_5^G will introduce terms modifying the discretization of D_{zz}. The sum of these two modifications is

$$2\left(\frac{s_7s_8}{(s_1+s_4)s_4}-\frac{s_9s_{12}s_7s_8}{(s_1+s_4)s_4(s_2+s_5)s_5}+\frac{s_{10}s_{11}}{s_2(s_2+s_5)}\right)$$
$$\times\left(\frac{1}{s_3s_6}f_0-\frac{1}{(s_3+s_6)s_3}f_3-\frac{1}{(s_3+s_6)s_6}f_6\right).$$

It must be shown that the coefficient is less than one. To do this, note that by neglecting the negative term and the s_1 and s_2 terms, the coefficient is smaller than

$$\frac{s_7s_8}{s_4^2}+\frac{s_{10}s_{11}}{s_5^2}.$$

Since $s_8 = s_4 - s_7$, the first term can be written as $s_7(s_4 - s_7)/s_4^2$ which achieves a maximum value of $1/4$ when $s_7 = s_4/2$. Similarly the second term can be shown to be less than $1/4$. Hence the total coefficient is less than $1/2$. It follows that the modifications to the D_{zz} terms are not enough to switch the overall sign of the matrix entries, and hence the linear system corresponding to the Poisson operator satisfies Eq. (8) for all rows. By considering Eq. (7) it can be seen that the inequality will be strict for any row where a Dirichlet condition is applied, since off-diagonal terms will be missing.

In addition to weak diagonal dominance, the discretized Poisson operator will in general satisfy the irreducibility condition [18]. Irreducibility can be defined by considering a directed graph where each node corresponds to a grid point, and for all i, j, if $A_{ij} \neq 0$ then there is an edge from node i to node j; the matrix is then said to be irreducible if for all i, j there is a path from node i to node j. Since the Poisson operator connects each node with all available orthogonal neighbors, it is reasonable to assume irreducibility will hold for any simply connected domain.

3 Multigrid Method on Non-Graded Cartesian Grids

3.1 General Multigrid Methods

Consider the problem

$$\begin{cases} Av = f & \text{in } \Omega, \\ v = v_0 & \text{on } \partial\Omega, \end{cases}$$

where A is a given linear operator. Let G^n be a grid on which the discretization of the previous problem can be written as

$$A_n v_n = f_n.$$

The multigrid method is an iterative approach that makes use of the simple Gauss–Seidel method as a component. Given an approximate solution u_n, several iterations of the Gauss–Seidel smoothing operator $S_n(u_n, f_n)$ will return an improved solution u_n'. The Gauss–Seidel method considers the local neighborhood of each grid point individually, and is efficient at damping high frequency errors. However, overall, the method is slow since it takes many iterations to remove low frequency errors, as information propagates slowly across the grid. The central idea behind the multigrid method is to upgrade u_n' by adding a suitable correction d_n, so that

$$A_n\big(u_n' + d_n\big) = f_n, \quad \text{or} \quad A_n d_n = r_n, \quad \text{where } r_n = f_n - A_n u_n' \text{ is the residual}$$

and to solve for the correction on a coarser grid G^{n-1}. On this grid, low frequency errors are more efficiently damped, and since the grid contains fewer points, it can be carried out rapidly.

To do this, a restriction operator R_n^{n-1} is needed to project the residual from G^n to G^{n-1}, after which a coarsened Gauss–Seidel operator S_{n-1} can be applied. On this grid, we make use of the zero solution as a starting guess, and solve for the coarsened residual $r_{n-1} = R_n^{n-1}(r_n)$ as a source term, so that

$$d_{n-1} = S_{n-1}(0_{n-1}, r_{n-1}).$$

We then make use of an interpolation operator I_{n-1}^{n} to compute $d_n = I_{n-1}^{n}(d_{n-1})$ on G^n, after which the solution on G^n can be updated according to $u'_n \rightarrow u'_n + d_n$. Following this, a number of additional Gauss–Seidel updates can be carried out on G^n.

This procedure is applied recursively, so that a hierarchy of grids $\{G^n, G^{n-1}, \ldots, G^0\}$ is introduced and residuals are recursively solved on coarser and coarser grids. The algorithm starts on G^n, and recursively solves residuals until reaching the coarsest grid G^0, after which the corrections are progressively applied up to G^n. Since this procedure goes down and up the hierarchy, it is referred to as a *V-cycle*. Multiple V-cycles can be applied until the desired level of accuracy is achieved.

To apply the Gauss–Seidel sweeps on the coarser grids, it is necessary to construct coarsened versions of the linear system A_n, and several approaches can be employed. In some situations, the discretization procedure used to construct A_n can also be used to construct the coarse problem. Alternatively, the coarse linear systems can be computed recursively by conjugating with the restriction and interpolation operators according to

$$A_{n-1} = R_n^{n-1} A_n I_{n-1}^n. \tag{11}$$

3.2 A Multigrid Method on Quadtree and Octree Grids

To implement the multigrid method on quadtree and octree grids, we must define a hierarchy of grids $\{G^n, G^{n-1}, \ldots, G^0\}$. An obvious choice of successive grids is simply the sequence of grids defined by the levels in the quadtree/octree itself. This choice may not be optimal, but has the advantage of being readily available [3]. We describe next the interpolation I_{n-1}^n and restriction R_n^{n-1} operators.

3.2.1 Trilinear Interpolation

Consider a vector u defined on G^k to be interpolated on the finer grid G^{k+1}. We use trilinear interpolation, defined for the unit cell $C = [0, 1]^3$ as

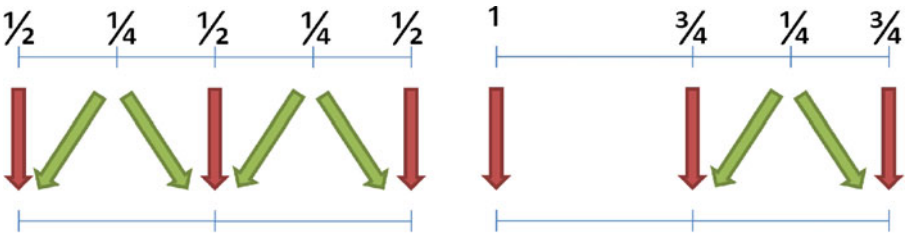
$$\begin{aligned} u(x, y, z) = {} & (1-x)(1-y)\big((1-z)u_{0,0,0} + zu_{0,0,1}\big) + (1-x)y\big((1-z)u_{0,1,0} + zu_{0,1,1}\big) \\ & + x(1-y)\big((1-z)u_{1,0,0} + zu_{1,0,1}\big) + xy\big((1-z)u_{1,1,0} + zu_{1,1,1}\big). \end{aligned}$$

Note that the interpolation procedure is used to interpolate the *errors* and one can simply define the errors outside the domain to be zero in order to treat all the points in Ω in the same fashion. This equation can now be used to define the interpolation operator I_k^{k+1} from G^k to G^{k+1}. One can easily verify that this operator is full-weighting since all its rows sum to one.

3.2.2 Full-weighting Restriction

To build the restriction operator R_{k+1}^k from G^{k+1} to G^k, we first define a possible restriction operator $\tilde{R}_{k+1}^k = c(I_k^{k+1})^T$ where c is a scaling factor (1/4 in 2D, 1/8 in 3D). For uniform grids, this operator is full-weighting as illustrated in Fig. 4(a). In the case of adaptive grids on the other hand, this property no longer holds because some grid points, and thus some coefficients in the restriction operator, are missing. Defining a non-full-weighting operator can have a drastic consequence on the convergence.

Therefore to ensure that our restriction is full-weighting we adjust the diagonal coefficients of $\tilde{R}_{k+1}^k$ such that all rows sum to one, and call R_{k+1}^k the resulting operator (see an illustration in Fig. 4(b)).



(a) Standard full-weighting restriction operator in the uniform case.

(b) Full-weighting restriction operator in the adaptive case.

Fig. 4 Full-weighting restriction operators. The values of the restricted function at every node of the coarser grid are defined using weighting coefficients depicted in (**a**) and (**b**)

3.2.3 Computation of the Matrices

In our computations, all of the operators A_k, R_{k+1}^k, and I_k^{k+1} are represented as sparse matrices with the compressed row format. The code begins by explicitly constructing A_n on the finest grid. Each grid point is considered sequentially, allowing the matrix to be assembled directly without the need for any reordering. The sequences of interpolations $I_{n-1}^n, \ldots, I_0^1$ and restrictions $R_n^{n-1}, \ldots, R_1^0$ are also directly constructed as described in Sects. 3.2.1 and 3.2.2, after which the coarsened linear systems $A_n, \ldots, A_0$ can be constructed using sparse matrix multiplication with Eq. (11). The time to carry out these operations scales linearly with the number of grid points, and is typically small compared to the time to carry out the V-cycles. We note that, unlike what is standard practice with multigrid methods, we do not maintain disjoint sets of interior versus boundary equations at each level of the hierarchy.

4 Numerical Results in Two Spatial Dimensions

In the work of [4], the above discretization has been shown to be second-order accurate in both the L^1 and L^∞ norms, thus we concentrate here on examining the computation time of our algorithm and how it scales with the number of grid points. Consider the problem

$$\begin{aligned} \Delta v &= \left(x^2 + y^2\right)e^{-xy} && \text{on } \Omega^-, \\ v &= e^{-xy} && \text{on } \Gamma. \end{aligned}$$

We consider for Ω^- two different domains represented by the level set functions $\phi_1(x, y) = \sqrt{x^2 + y^2} - 0.7$ and $\phi_2(r, \theta) = r - 0.4 + 0.1\cos(6\theta)$ as depicted in Figs. 5(a) and (b). Figures 6(a) and (b) give the time to convergence using a classical V-cycle method, defined as a succession of V-cycle iterations until the L^∞-norm of the normalized residual is smaller than a given tolerance ϵ. The normalized residual is obtained by dividing each coefficient of the residual by the corresponding diagonal coefficient in the discretization matrix and we use $\epsilon = 10^{-12}$ throughout this study. One Gauss–Seidel iteration is performed on the each level on the way down of the V-cycle, while five iterations are performed on the way up. The slopes of the best fit lines are 1.100 and 1.106 respectively for the two examples. This is close to the theoretical complexity of $O(N)$ and the small amount above unity is likely to be due to lower cache efficiency for the larger grids with more memory allocation.

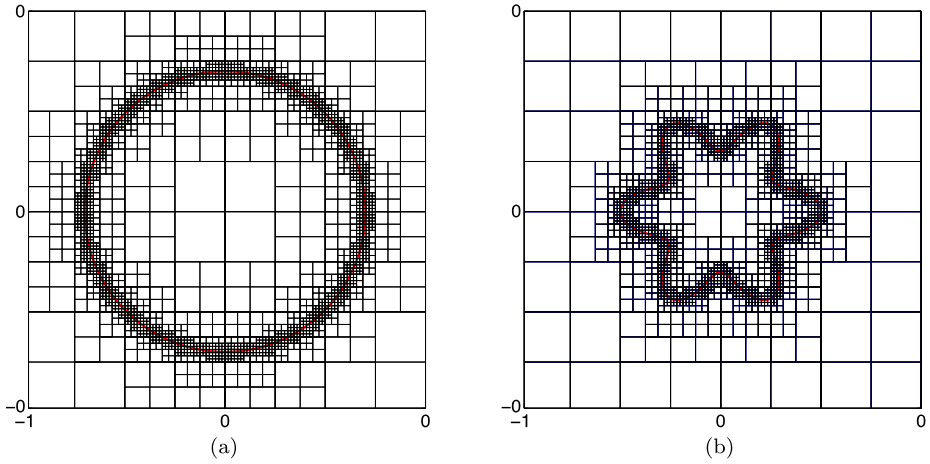


Fig. 5 Contours of the irregular domains (*red*) considered in Sect. 4. The grids (*black*) correspond to a level 7 Quadtree

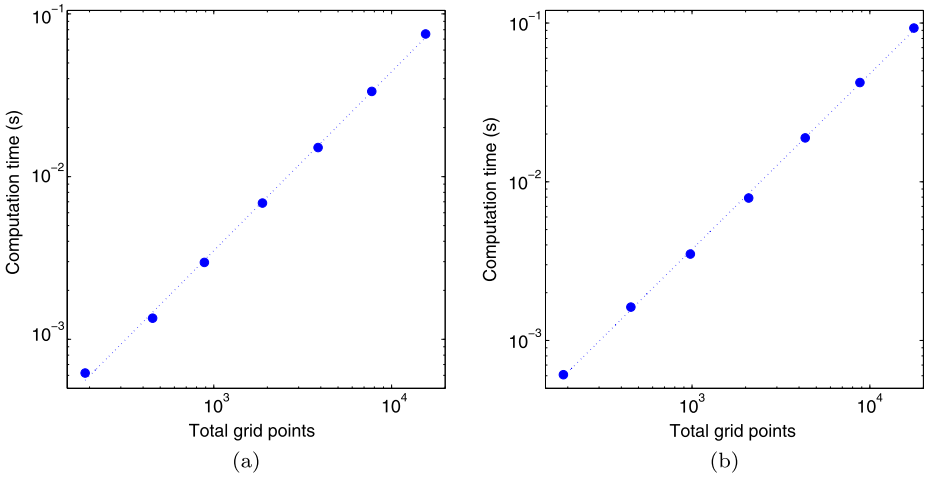


Fig. 6 Computation times in a log-log scale for the multigrid algorithm, using different levels of refinement for the two examples considered in Sect. 4. (**a**) Corresponds to the irregular domain depicted in Fig. 5(a) while (**b**) corresponds to the irregular domain depicted in Fig. 5(b)

4.1 Convergence Rates

The convergence properties of the method for the two test problems are shown in Figs. 7(a) and (b). The calculations are performed on level 10 quadtree grids containing 15 537 and 17 689 grid points respectively. For the comparison, the convergence of the classical Bi-CGSTAB and for the Gauss–Seidel methods are represented. The plots show the convergence of the multigrid algorithm with the full-weighting restriction, and also show the convergence of the multigrid algorithm when the restriction is taken to be the transpose of the interpolation operator. It is clear that the multigrid algorithm using the non-full-weighting rule requires significantly more iterations to converge than the approach using the full-weighting

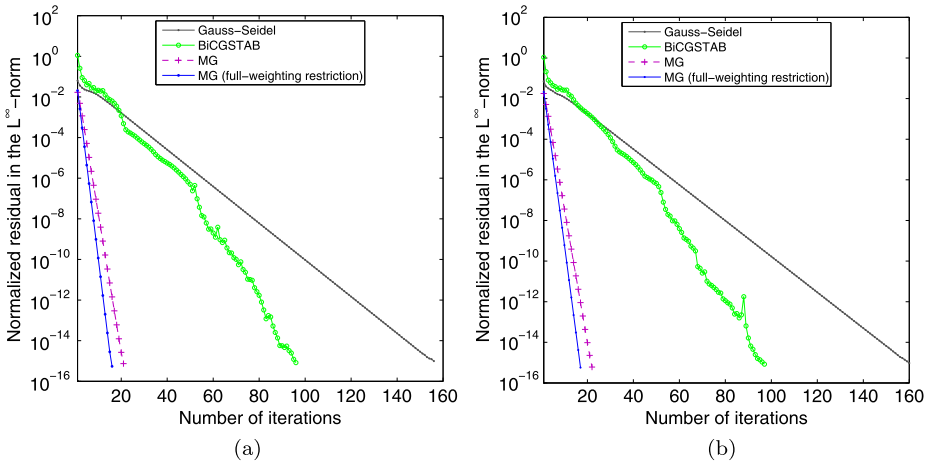


Fig. 7 Normalized residual versus the number of iterations for the examples considered in Sect. 4. (**a**) Corresponds to the irregular domain depicted in Fig. 5(a) while (**b**) corresponds to the irregular domain depicted in Fig. 5(b). The *purple line* that is labeled "MG" corresponds to the non-full-weighting multigrid algorithm

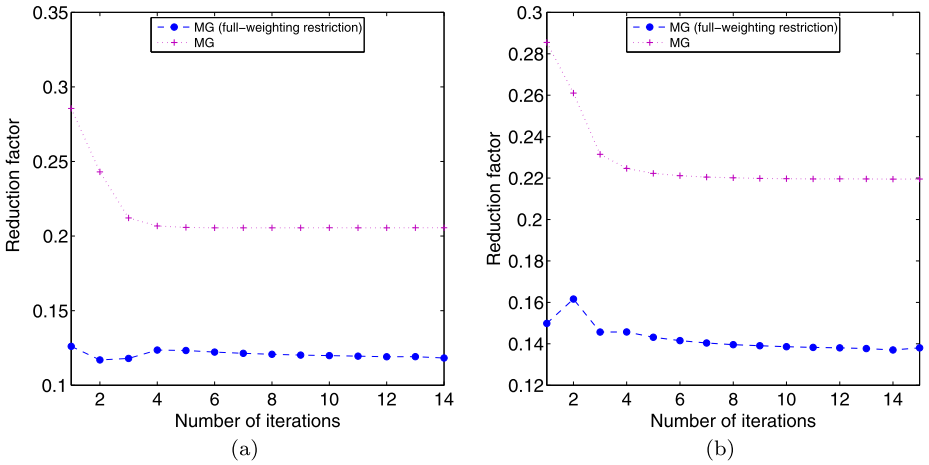


Fig. 8 Reduction factor for the examples considered in Sect. 4. (**a**) Corresponds to the irregular domain depicted in Fig. 5(a) while (**b**) corresponds to the irregular domain depicted in Fig. 5(b). The *purple line* that is labeled "MG" corresponds to the non-full-weighting multigrid algorithm

rule. The resulting reduction factor, defined as the ratio between the residual at two consecutive iterations, is depicted in Figs. 8(a) and (b). It can be seen that the non-full-weighting restriction has reduction factors that are roughly twice as large.

It should be noted that both Bi-CGSTAB and Gauss–Seidel also perform well on the test problems considered. This should be expected, since on the adaptive grids, the bulk of the grid points are close to the boundary where the Dirichlet boundary condition is applied, and the solution at these points is somewhat easier for any linear solver to find. The test problems considered feature a minimal amount of refinement concentrated on the boundary where the Dirichlet condition is applied, which is almost the best case for these methods. Thus, while

the multigrid algorithm is faster, the improvement over the other methods is not as dramatic as on uniform grids. It is reasonable to expect that for more complex problems, with more refinement in regions away from where Dirichlet conditions are imposed, the advantages of multigrid will become more apparent. Indeed, it is worth noting that a uniform grid can be represented within a quadtree data structure, and for this case the algorithm is equivalent to a standard uniform multigrid computation, and will therefore have the same convergence properties.

4.2 Impact of the Boundary Condition Strength Coefficient

When assembling the matrix A_n on the finest level, we impose the arbitrary value of $u = 0$ for all the nodes in Ω^+. This is achieved by setting the diagonal element to BC_{strength}, the extra-diagonal elements to be zero and the right-hand-side to be 0. In the case where the interface is less than a small tolerance ϵ_D away from a grid node, we simply impose the boundary condition at that node, by setting the diagonal element to BC_{strength}, the extra-diagonal elements to be zero and the right-hand-side to be $BC_{\text{strength}} \times \gamma$, where γ is the interface boundary condition. Throughout this study, we make use of $\epsilon_D = \Delta x^2$. In the calculations presented thus far, BC_{strength} was set to 1.

For a simple Gauss–Seidel method on the finest grid, the value of BC_{strength} plays no role, since it only serves to enforce solution values at certain grid points. However, in the multigrid method BC_{strength} has an effect, since the rows in coarse matrices A_i are constructed as averages of rows in the fine matrix. Close to the interface, a coarse matrix row will be assembled as a linear combination of rows where the values are imposed, and rows where the operator discretization is used. The value of BC_{strength} therefore controls how strongly the boundary conditions are felt on the coarser levels. We have found that this parameter can have a strong effect on the rate of convergence, and in this section we present results for a range of BC_{strength}. To aid in comparisons between grids with differing resolution, we impose that all matrix rows where boundary conditions are not imposed are rescaled so that the diagonal entry is one.

Figures 9(a) and (b) show the decrease in the normalized residual for different values of BC_{strength}. While we have observed that the method converges over a very large range of BC_{strength}, the choice of this parameter can alter the convergence rate by a factor of two or more. Numerical tests indicate that the optimal value of BC_{strength} is approximately 0.125, since for both bigger and smaller values more iterations are needed to converge to the same residual.

4.3 Convergence Analysis

Figures 7(a) and (b) show that the algorithm converges rapidly for the test problems, but we now consider the convergence properties of the algorithm in general. While multigrid convergence for symmetric problems has been extensively studied [19], there are fewer analytic results for non-symmetric systems like the ones that we consider. We therefore do not present a complete proof of convergence for all cases, but instead analyze individual components of the algorithm to provide confidence that the method would work across a wide range of problems.

As noted in Sect. 2.4, the linear system on the finest level is weakly diagonally dominant, and it is also reasonable to assume that it is irreducible, in which case it is known that the Gauss–Seidel method will converge for any ordering of grid points [18].

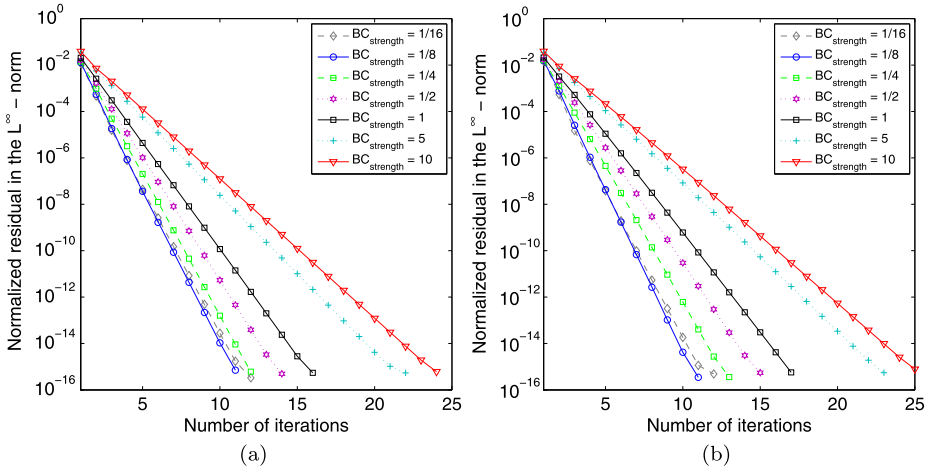


Fig. 9 Influence of the boundary condition coefficient on the convergence of the multigrid method. (**a**) Corresponds to the irregular domain depicted in Fig. 5(a) while (**b**) corresponds to the irregular domain depicted in Fig. 5(b)

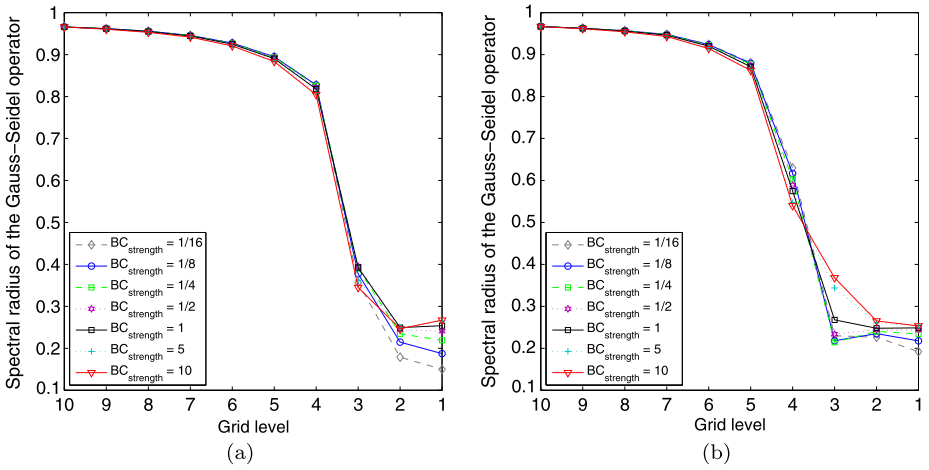


Fig. 10 Spectral radii of the Gauss–Seidel operators on each level for the examples presented in Sect. 4. (**a**) Corresponds to the irregular domain depicted in Fig. 5(a) while (**b**) corresponds to the irregular domain depicted in Fig. 5(b)

On the coarser levels, where the linear systems are computed according to Eq. (11), establishing weak diagonal dominance cannot be carried out easily. However, for the two test problems, we have explicitly checked that the coarse systems are diagonally dominant. In addition, Figs. 10(b) and (a) show explicit computations of the spectral radii for the Gauss–Seidel operators on each level for a variety of values of BC_{strength}, showing that they are all below one, and become particularly small on the coarsest grids. The differences due to BC_{strength} are more pronounced on the coarser grids, and the trends appear to be roughly consistent with the convergence properties seen in Figs. 9(a) and (b).

Knowing that all the spectral radii of the Gauss–Seidel operators are smaller than one still does not guarantee convergence, since it is possible for errors to be multiplied in the restriction and interpolation steps. As a final check, for the two example problems with a level 8 quadtree grid, we explicitly computed the spectral radii of the V-cycle operators and found them to be 0.1015 and 0.1319 respectively. As would be expected, these values are close to the reduction factors seen in Figs. 8(a) and (b). However, since the spectral radius computation involves finding the maximum eigenvalue of the V-cycle operator, it allows us to make a stronger statement that convergence will be guaranteed for any initial guess.

5 Numerical Results in Three Spatial Dimensions

We consider the problem

$$\begin{aligned} \Delta v &= \left(x^2z^2 + y^2z^2 + x^2y^2\right)e^{-xyz} \quad \text{on } \Omega^-, \\ v &= e^{-xyz} \quad \text{on } \Gamma, \end{aligned}$$

where Ω^- is a sphere defined on $\Omega = [-1, 1]^3$, centered at the origin with radius $r = 0.7$. The boundary condition and the right hand side are chosen such that the exact solution is $v = e^{-xyz}$. Figure 11(a) depicts the contour obtained using a level 6 octree grid. The time needed to converge for various levels of refinement is depicted in Fig. 11(b), comparing the multigrid method to Bi-CGSTAB. The slope of the best fit line for the multigrid is 1.137, giving again a scaling close to the theoretical computation. The slope of the best fit line for the Bi-CGSTAB is 1.199, which outperforms the usual $O(N^{5/4})$ convergence for the same reasons as those discussed in the two-dimensional case of Sect. 4.1. Even though the two slopes are similar, the computation time is approximately four times smaller for the multigrid, making this method more efficient.

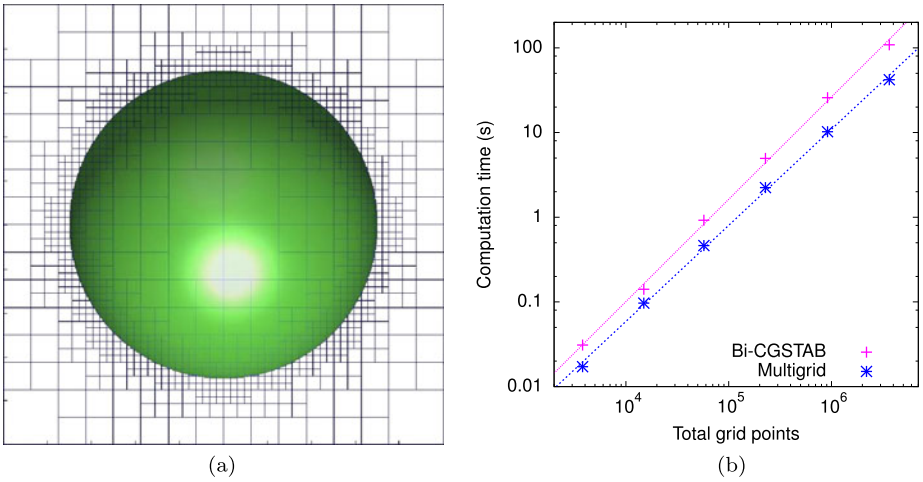


Fig. 11 (**a**) Contour the corresponding grid for the example considered in Sect. 5. Only a two-dimensional cross section of the grid is plotted. (**b**) Corresponding computation times in a log-log scale for the multigrid algorithm, using different levels of refinement. A comparison to the classical Bi-CGSTAB method is shown. For each plot, *solid lines* represent best linear fits

6 Conclusion

We have presented the results of a multigrid solver for the Poisson equation on irregular domains on non-graded adaptive quadtree and octree grids. Dirichlet boundary conditions are imposed at the irregular domain's boundary. The use of non-graded adaptive quadtree and octree data structures significantly reduces the number of computational nodes and allows for the resolution of small length scales. The development of a multigrid method on such grids further accelerates the convergence process, producing an efficient approach that is particularly suitable for multiscale problems. We have presented numerical results that illustrate the efficiency of such an approach.

Acknowledgements M. Theillard and F. Gibou were supported by the Department of Energy under contract number DE-FG02-08ER15991; by the Office of Naval Research through contract N00014-11-1-0027; the National Science Foundation through contract CHE-1027817; the W.M. Keck Foundation; and by the Institute for Collaborative Biotechnologies through contract W911NF-09-D-0001 from the U.S. Army Research Office. C.H. Rycroft was supported by the Director, Office of Science, Computational and Technology Research, U.S. Department of Energy under contract number DE-AC02-05CH11231.

References

1. Chen, H., Min, C., Gibou, F.: A supra-convergent finite difference scheme for the Poisson and heat equations on irregular domains and non-graded adaptive Cartesian grids. J. Sci. Comput. **31**(1), 19–60 (2007). doi:10.1007/s10915-006-9122-8
2. Berger, M., Oliger, J.: Adaptive mesh refinement for hyperbolic partial differential equations. J. Comput. Phys. **53**, 484–512 (1984). doi:10.1016/0021-9991(84)90073-1
3. Popinet, S.: Gerris: a tree-based adaptive solver for the incompressible Euler equations in complex geometries. J. Comput. Phys. **190**, 572–600 (2003). doi:10.1016/S0021-9991(03)00298-5
4. Min, C., Gibou, F., Ceniceros, H.: A supra-convergent finite difference scheme for the variable coefficient Poisson equation on non-graded grids. J. Comput. Phys. **218**, 123–140 (2006). doi:10.1016/j.jcp.2006.01.046
5. Chen, H., Min, C., Gibou, F.: A numerical scheme for the Stefan problem on adaptive Cartesian grids with supralinear convergence rate. J. Comput. Phys. **228**(16), 5803–5818 (2009). doi:10.1016/j.jcp.2009.04.044
6. Min, C., Gibou, F.: A second order accurate projection method for the incompressible Navier–Stokes equation on non-graded adaptive grids. J. Comput. Phys. **219**, 912–929 (2006). doi:10.1016/j.jcp.2006.07.019
7. Min, C., Gibou, F.: A second order accurate level set method on non-graded adaptive Cartesian grids. J. Comput. Phys. **225**, 300–321 (2007). doi:10.1016/j.jcp.2006.11.034
8. Brandt, A.: Multi-level adaptive solutions to boundary-value problems. Math. Comput. **31**, 333–390 (1977). doi:10.1090/S0025-5718-1977-0431719-X
9. Briggs, W., Henson, V.E., McCormick, S.: A Multigrid Tutorial, 2nd edn. SIAM, Philadelphia (2000). ISBN 0-89871-462-1
10. Sampath, R.S., Biros, G.: A parallel geometric multigrid method for finite elements on octree meshes. SIAM J. Sci. Comput. **32**(3), 1361–1392 (2010). doi:10.1137/090747774
11. Haber, E., Heldmann, S.: An octree multigrid method for quasi-static Maxwell's equations with highly discontinuous coefficients. J. Comput. Phys. **223**(2), 783–796 (2007). doi:10.1016/j.jcp.2006.10.012
12. Osher, S., Sethian, J.: Fronts propagating with curvature-dependent speed: algorithms based on Hamilton–Jacobi formulations. J. Comput. Phys. **79**, 12–49 (1988). doi:10.1016/0021-9991(88)90002-2
13. Osher, S., Fedkiw, R.: Level Set Methods and Dynamic Implicit Surfaces. Springer, New York (2002)
14. Sethian, J.: Level Set Methods and Fast Marching Methods. Cambridge University Press, Cambridge (1999)
15. Strain, J.: Fast tree-based redistancing for level set computations. J. Comput. Phys. **152**, 664–686 (1999). doi:10.1006/jcph.1999.6259
16. Min, C.: Local level set method in high dimension and codimension. J. Comput. Phys. **200**, 368–382 (2004). doi:10.1016/j.jcp.2004.04.019
17. Samet, H.: Applications of Spatial Data Structures: Computer Graphics, Image Processing and GIS. Addison-Wesley, New York (1990)
18. Demmel, J.W.: Applied Numerical Linear Algebra. SIAM, Philadelphia (1997)
19. Yserentant, H.: Old and new convergence proofs for multigrid methods. Acta Numer. **2**, 285–326 (1993). doi:10.1017/S0962492900002385