A geometric multigrid method based on L-shaped coarsening for PDEs on stretched grids

H. bin Zubair*[1], S. P. MacLachlan[2], and C. W. Oosterlee[3]

[1] *Numerical Analysis Group, DIAM, Faculty EEMCS, Delft University of Technology, Mekelweg 4, 2628 CD, Delft, The Netherlands, Email: h.binzubair@tudelft.nl*

[2] *Department of Mathematics, Tufts University, Bromfield-Pearson Building, 503 Boston Avenue, Medford, MA 02155, USA. scott.maclachlan@tufts.edu*

[3] *'Centrum voor Wiskunde en Informatica (CWI), Dutch national research centre for mathematics and computer science' Modelling, Analysis and Simulation (MAS2) Kruislaan 413, Amsterdam Email: c.w.oosterlee@cwi.nl*

SUMMARY

In this work, we present a geometric multigrid method for PDEs discretized on stretched grids. The emphasis is on geometric L-shaped coarsening techniques that we have developed in this context. The presented method is matrix free, in contrast with alternatives such as Algebraic Multigrid (AMG) or certain preconditioned Krylov-subspace based solution methods. For a Poisson model problem, we explain, both visually and in a descriptive way, how the stretched fine grid may yield a sequence of coarser grids so as to maintain the complementarity between relaxation and coarse-grid correction. We also present complexity estimates of the method, thus demonstrating its efficiency. Through figures and numerical experiment tables, we provide convergence histories for the model problem discretized -and solved- on various stretched grids with our method.

KEY WORDS: Matrix-free multigrid, Grid stretching, L-shaped coarsening techniques, Cell-centered FVM discretization.

1. Introduction

Many real-life application problems have discontinuities or kinks at specific regions of the domain and, therefore, the selection of a discretization grid is usually dictated by specific accuracy requirements in these regions. The regions are often restricted locally to certain parts of the domain, thereby requiring a higher concentration of grid points there. This saturation of grid points in certain areas of the domain gives rise to anisotropy in the resulting linear system. This anisotropy -due to strongly varying connection strengths in the discrete operator- poses well-known convergence problems for standard multigrid with point-wise smoothing [1, 2]. There are two broad categories of remedy for this issue; either keep standard coarsening and modify the smoother (using block-smoothing, for example), or keep point-wise smoothing and modify the coarsening in a way so that complimentarity

*Correspondence to: H. bin Zubair; Email: h.binzubair@tudelft.nl, h.binzubair@gmail.com

Contract/grant sponsor: This research is supported in parts by three sponsors. (1) The Dutch government through the national program BSIK: knowledge and research capacity, in the ICT project BRICKS (http://www.bsik-bricks.nl), theme MSV1, (2) The Government of Pakistan through The HEC-Pakistan's research grant, contract Ref: 1-3/PM-OVER/Neth/SPMU/2004. (3) The European Community's Sixth Framework Programme, through a Marie Curie International Incoming Fellowship, MIF1-CT-2006-021927. We would like to express our thanks to all of them.

between the smoothing process and the coarse-grid correction process is preserved. The former approach is well-developed and precisely known [1, 3], although its practical use is not very viable when problem dimensionality increases. The latter approach is open to development and numerous works have surfaced in this context, see [4, 5, 6, 7].

Higher grid resolution may be obtained from two geometrically different ideas, *AMR* (*Adaptive Mesh Refinement*) [8, 9, 10, 11] and *Grid Stretching*; both approaches may give rise to *structured* and *unstructured* grids. For geometric classification, we refer the grids arising from locally adapted mesh refinement as *amr-type* grids, while those resulting from the use of a global stretching parameter (detailed in Section 2.3) as *str-type* grids. Structured *str-type* grids form the main theme in this paper. Solution methods for unstructured grids usually make use of *quad* -or *oct* tree data structures [12, 13]. *Uniform* grid stretching occurs when mesh sizes are equidistant throughout a particular dimension but are non-equidistant across different dimensions. On the contrary, *non-uniform* grid stretching can be defined as the case where the grid has variable mesh sizes even within a single dimension. These grids are often the result of a coordinate transform of the grid variables.

In [14], we showed that for uniformly stretched grids, partial coarsening along the stretched dimensions gave an optimal multigrid algorithm. In this work, we consider non-uniformly stretched grids for a two-dimensional model problem. The grid-stretching that we treat in this paper is called Power-law grid stretching. The domain is rectangular, and the grid is Cartesian, which lends a specific structure to be exploited for efficiency.

A well known alternative to geometric multigrid treatment of grid stretching is the use of algebraic multigrid (*AMG*) algorithms [15, 16]. This falls in the category of solution techniques employing point-based relaxation and seeking to coarsen the grid in a way so that errors on the coarse grid appear geometrically smooth. AMG starts out with a given matrix (embodying the sparse algebraic equations) and constructs all of the multigrid components algebraically, through variational principles, during the actual solution process. This entails a setup phase (which, consequently, has a cost attached to it) in which these components are set up, in the form of matrix operators, for use during the iterative solution phase. The capability to construct multigrid components straight from the algebraic equations without knowledge of the discretization grid renders AMG very robust. AMG is, thus, a solution method of choice, whenever the geometric counterpart is too difficult (or impossible) to apply due to the complex underlying geometry of the problem. However, the price of this robustness is an extra cost both in terms of storage as well as computation. Ironically, the very robustness of AMG comes at the price of compromising any geometric structure that the given problem might have to offer, and which, in turn, might be exploited for efficiency and better storage costs. *Naturally then, for problems that are geometrically tractable, the aim should be to achieve the excellent multigrid convergence that AMG has to offer without compromising the benefits of the geometrical structure in the discrete problem.*

In this work, we present one method of circumventing the drawbacks of AMG while still ensuring its excellent convergence factors for PDEs on stretched grids. We introduce a hybrid technique that uses a coarsening pattern inspired by AMG, in a geometric multigrid setting. This coarsening pattern for the stretched grids under consideration is along so-called *L-shaped* lines. The key idea in this work is to gradually relax grid-anisotropies by employing this coarsening pattern, in a completely geometric multigrid method based on point smoothing. This allows us to exploit the structure that a square domain and a Cartesian grid have to offer. We combine this L-shaped coarsening with point smoothing, piece-wise constant restriction and bi-linear interpolation and present a geometric method that converges very well for the Poisson model problem.

An outline of the paper is as follows. In Section 2, we specify the model problem, the Power-law grid-stretching scheme, and the cell-centered finite-volume discretization that we use. Section 3 follows, with details of the geometric multigrid components for the new method. Here, we explain in precise detail the L-shaped grid-coarsening technique, along with a visual display of the coarsened grids obtained. The discretization-coarse-grid (*DCG*) operator (see [3]) comes next, and is followed by the description of the transfer operators. In Section 4, we do a complexity analysis for this method. This is followed by Section 5, in which we provide numerical experiments based on 1d and 2d grid stretching. A simple jump discontinuity experiment and an experiment on *amr-type* grids are also performed to test the robustness of the method. Convergence histories are presented both in tabular and in visual displays; finally, in the last section, some conclusions are drawn from the work. An appendix follows, where we derive the flux balance equations completely for two example cases, thus demonstrating how we carry out coarse-grid correction in practice.

2. The model problem, the cell-centered FVM, and grid stretching

2.1. The model problem

We have chosen a 2d scalar Poisson-type equation as our model problem. It provides a consistent model upon which new numerical solvers can be tested without delving too deeply into the complications of many application problems. This two-dimensional Poisson-type equation (on a unit square, with Dirichlet boundary conditions) is given as:

$$\begin{aligned} -\nabla \cdot \mathbf{A}\nabla u(x,y) &= f^{\Omega}(x,y), \quad (x,y) \in \Omega = (0,1)\times(0,1) \\ u(x,y) &= f^{\Gamma}(x,y), \quad (x,y) \in S \Rightarrow x \in \{0,1\} \text{ or } y \in \{0,1\} \end{aligned} \tag{1}$$

where

$$\mathbf{A} = \begin{bmatrix} a_1(x,y) & 0 \\ 0 & a_2(x,y) \end{bmatrix}$$

and a_1 and a_2 are smoothly varying functions.

2.2. The cell-centered finite-volume scheme

To define a cell-centered finite-volume scheme, the entire domain is divided into rectangular control volumes, with nodes in the center of the cells. Each node represents the value of the unknown averaged over the control volume. Therefore, after integrating both sides of Equation (1), and then applying the Gauss Divergence theorem, the result can be written in the form of the following sum,

$$\sum_{m} \Big\{ -\sum_{k=0}^{3} \int_{S_{m_k}} (\mathbf{A}\nabla u_{m_k}) \cdot \hat{n}_k \, dS_{m_k} = \int_{\Omega_m} f^{\Omega} d\Omega_m \Big\}, \tag{2}$$

where m indexes a rectangular control volume, S_{m_k} denotes the k^{th} face of the boundary of control volume m, u_m is the value of u averaged over the control volume, m, and (∇u_{m_k}) refers to the gradient of u_m computed at the midpoint of S_{m_k}. Ω_m is the volume of the m^{th} cell, such that $\Omega = \cup\, \Omega_m$. Equation (2) is satisfied by ensuring that Eqn. (3) holds for all m,

$$-\sum_{k=0}^{3} \int_{S_{m_k}} (\mathbf{A}\nabla u_{m_k}) \cdot \hat{n}_k \, dS_{m_k} = \int_{\Omega_m} f^{\Omega} d\Omega_m \tag{3}$$

Eqn. (3) is the foundation equation of FVM here, which can be approximated in a number of ways, one of which is detailed in Section 3. When this is done for all values of m, the result is an $(m \times m)$ system of linear equations.

2.3. Grid stretching and the Power-law scheme

In many application problems, grid stretching is motivated by the need of greater accuracy requirements in specific portions of the domain.

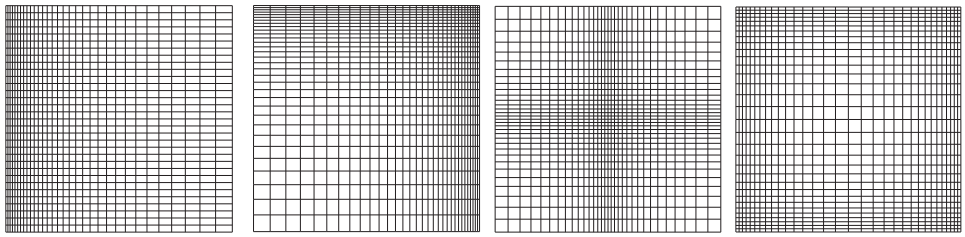
- In [17], for example, the diffusive Maxwell equations on very large $3D$ domains are investigated. The particular accuracy requirements in this work force the use of a stretched Cartesian grid. The stretching is brought about with the Power-law scheme. In this paper, multigrid convergence, with point-wise smoothing and standard coarsening, deteriorates significantly on these grids.
- It is well-known that the convection-diffusion equation with Dirichlet boundary conditions, which is often used in modelling the flow during reservoir-fills, may give rise to steep boundary layers (see [1], Section 7.1). For an accurate simulation, the grid density has to be substantially higher in these regions, leading to a stretched grid with unknowns saturated around the boundary layers.
- For computing the flow around an airfoil, stretched grids, often referred to as c-grids, are used. (See [1], Section 9.6)
- Scattering applications in quantum mechanics [18] may require transformation of the Schrodinger equation to a more tractable Helmholtz-type equation. The wavenumber in this application is spatially dependent and grows strongly at the boundary. This gives rise to the so-called evanescent waves in the solution, which may require L-shaped grid concentration near the boundaries.

This emphasizes that efficient multigrid methods on stretched grids are required and, therefore, treated in this paper. We stay, however, with model problem experiments on Cartesian grids and focus on a novel coarsening technique.

In this work, we treat multigrid for a specific variety of *str-type* grid-stretching, called the Power-law grid stretching. In what follows, we use the terms *left* and *right* to indicate the decreasing and the increasing directions -respectively- along a particular dimension. x_{str} is the point from where the stretching ensues. Dirichlet boundaries are pinned down first, and then we divide the segment to the left and the right of the stretching point into a specified number of control volumes. The mesh sizes along each dimension are generated by the 1-dimensional formula presented in Eqn. (4). In the following, x_{min}, x_{max} are the domain boundaries along the i^{th} dimension, so that $x_{min} \leq x_{str} \leq x_{max}$ holds. N_L and N_R are the specified number of cells on the left and the right of x_{str}, respectively. Similarly α_L and α_R are the stretching parameters -to be used- respectively on the left and the right of x_{str}. When this is provided, we choose:

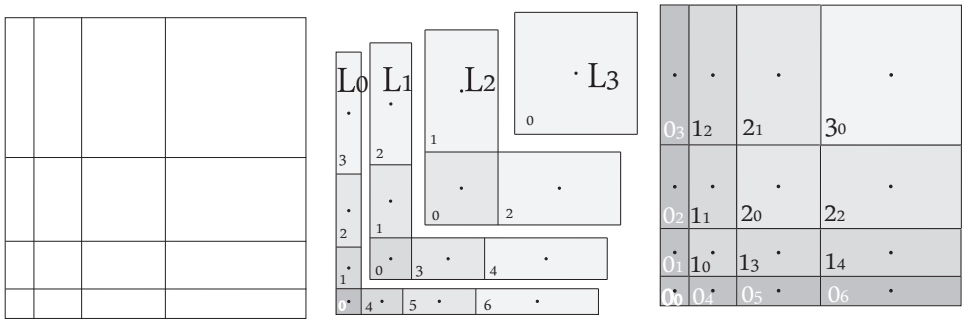
$$\begin{aligned} h_k &= \left\{ (x_{str} - x_{min}) / \left(\frac{\alpha_L^{N_L} - 1}{\alpha_L - 1} \right) \right\} \times \alpha_L^{(N_L - 1 - k)}, \quad k = 0 \cdots (N_L - 1); \\ h_k &= \left\{ (x_{max} - x_{str}) / \left(\frac{\alpha_R^{N_R} - 1}{\alpha_R - 1} \right) \right\} \times \alpha_R^{(k - N_L)}, \quad k = N_L \cdots (N_L + N_R - 1); \end{aligned} \tag{4}$$

Using this, we can generate a host of stretched grids, some of which are depicted in Figure 1. In Section 5, we experiment both with one -and two -dimensional grid stretching. For two-dimensional stretching, the method that we present in this paper has been developed for grids having control volume concentration at a



(a) West edge (b) North east corner (c) Center (d) All corners

Figure 1. (*str-type* grids): A depiction of different *str-type* concentrations of control volumes, obtained through (4) with $\alpha = 1.07$



(a) A 4×4 stretched grid (b) Display of L-strips (c) Control volume ordering

Figure 2. Grid enumeration in L-shaped strips

domain corner. However, we point out that the other two-dimensional stretchings (i.e., those having concentrations either at the center or at all the four corners of the domain) are merely unions of the types that we treat here and, therefore, the results carry over to them as well.

3. Multigrid with L-shaped grid coarsening, the DCG, restriction, and prolongation operators

3.1. The enumeration-scheme and L-shaped coarsening

In this paper, we focus on domains that are stretched with the same parameter for both x and y dimensions (i.e., $\alpha = \alpha_x = \alpha_y$). This allows us to use symmetry advantageously for storage purposes. The entire domain is divided into L-shaped control-volume strips. In general, an L-strip consists of a *vertex-cell*, a *vertical segment* consisting of control volumes, and likewise, a *horizontal segment*. The different L-strips in the discrete domain have different number of cells, each of which has a different volume. The different L-strips are ordered in priority of cell density, so that the most densely saturated strip comes first, and the strips with descending saturation of control-volumes follow. The last strip invariably consists of only the vertex control volume, being devoid of horizontal -and vertical segments.

Example 1. (Enumeration of the L-strips and ordering of the cells) *Consider Figure 2. (a) shows a simple stretched Cartesian grid with* 4×4 *control volumes. In (b), these control volumes are enumerated into the* 4 *L-strips,* $L_0 \cdots L_3$*, and nodes are placed in the center of each control volume. The dissectioning is virtual and only shown as a depiction of the enumeration. Note the ordering of the control volumes. Each L-strip complies with*

the general description in this section. (c) shows the same grid as in (a) but with the grid enumerated in L-strips and control volumes carrying the described ordering. In the cell-indices in (c), the larger digit represents the index of the L-strip, while the smaller ones represent the indices of the control volumes within it.

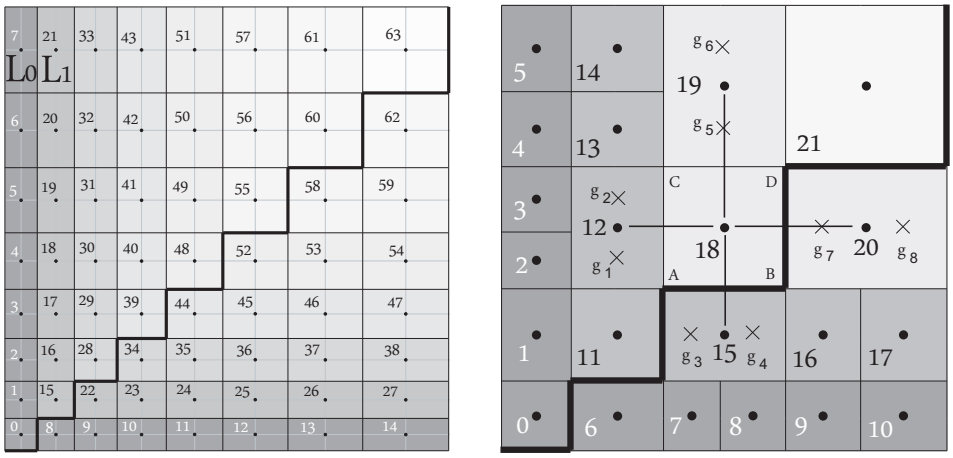
Grid coarsening in this setting is performed by agglomeration of control volumes on the fine grid [19, 20]. Selection of prospective fine-grid cells to agglomerate is done by virtually isolating two L-lines and comparing the different mesh aspect ratios that we seek to improve through the coarsening process. The mesh aspect ratio, *mar*, of a cell is defined as:

$$mar = \frac{h}{w},$$

where h and w represents the height and the width of the cell, respectively.

The global guiding principle in selecting fine-grid cells to agglomerate is that the newly constructed coarse-grid cell should reflect an improvement in the mesh-aspect ratio over other prospective fine-cell agglomerations. A pair of L-strips is virtually isolated and inspected cell by cell. The process is guided by the enumeration scheme of our method. Coarsening starts by visiting the cells near the vertex and agglomerating them. This agglomeration always gives a perfectly square coarse cell. Then the vertical segment is inspected, 4 cells at a time, i.e., 2 cells of each adjacent vertical strip; and a choice is made between horizontal-semi-coarsening (2×1) or full-coarsening (2×2). This choice is not solely dependent on the mesh aspect ratios; in fact, it is biased in favour of full coarsening if this comes within limits of a particular bound, SQTOL, that we set experimentally. The decisions are stored and automatically carried over to the horizontal segment due to symmetry. A pair of fine-grid L-strips thus gives a coarse-grid L-strip. This process of inspecting fine-grid L-strips in pairs is continued until the grid is depleted. It is important to note that this particular coarsening process leads us to store a *connection structure* which has exactly half the number of elements as the control volumes in the given grid. The following example is provided to elaborate on this process in greater detail.

Example 2. (L-shaped coarsening) *Consider the grids in Figure 3. The grid in Figure 3(b) represents the coarse grid chosen for the fine grid in Figure 3(a). To illustrate the process, we inspect L_0 and L_1 together, in Figure 3(a). First, we agglomerate the vertex cells,* $0, 1, 8, 15$, *and then move up the vertical segment. Two different prospective agglomerations to consider next are either combining Cells* $2, 3, 16, 17$ *together, called prospect 1, or combining Cells* 2 *and* 16*, called prospect 2. The coarsening pattern that we employ is such that individually agglomerating Cell* 2 *with Cell* 3 *and Cell* 16 *with Cell* 17 *is not an option, nor is not agglomerating at all. Put simply, we rule that while traversing the vertical segment, the decision has to be made between semicoarsening in the x-direction or full-coarsening, depending on mar_1 and mar_2, which are the mesh aspect ratios of the two propects, respectively. This ensures that nodes stay aligned along* 1 *dimension and reduces unnecessary book-keeping. We define the difference, d_i, for the i^{th} coarsening prospect as $d_i = |1 - mar_i|$. Although it seems relatively simple to pick the prospect with the smaller difference, we point out that this does not lead to an optimal reduction of complexity. As mentioned earlier, we set a priority criterion in favour of prospect 1, as it allows a greater reduction of unknowns, compared with prospect 2. We choose a value, SQTOL, such that if $d_1 \leq SQTOL$ then prospect 1 is selected and prospect 2 dropped, even if $d_2 < d_1$; however, if comparison with SQTOL fails, then the prospect with the lesser d is selected straight away. On the coarser grids, there is an additional check; if a situation such as depicted by Cells* $2, 3, 12$ *of Figure 3(b) arises, then we simply carry out this 3-cell agglomeration and do not venture to fatten the prospective coarse-grid cell any further. In particular, this check*



(a) (b)

Figure 3. (a) shows an (8×8) grid concentrated at the lower-left corner. The stretching parameter used along both dimensions is $\alpha = 1.07$ and $x_{str} = 0.0$. Thin lines connecting the nodes emphasize that -unlike coarser levels- all nodes on the finest grid are aligned horizontally and vertically (there are no hanging nodes). The heavy stair-case line shows the symmetry of the grid which is advantageous both from a storage as well as a computational point of view. (b) displays the first coarse-grid constructed from (a) using the technique described in this section. Note that there may be hanging nodes on the coarser levels.

ensures that the boundary of the coarsened cell is shared with its neighbour in the adjacent L-strip.

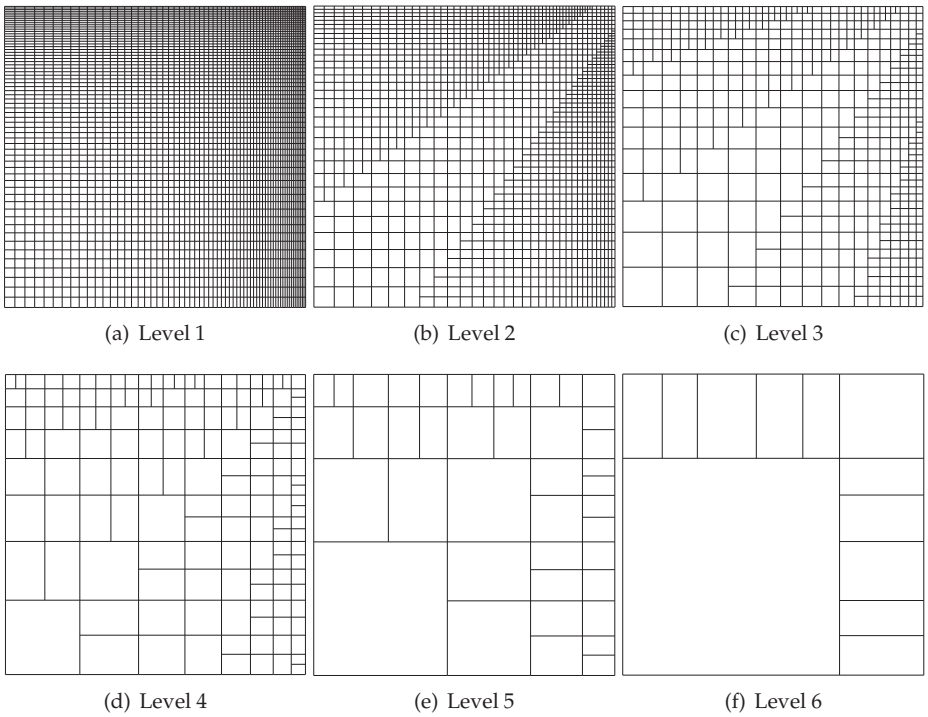
On the finest grid, the coarsening process detailed above results automatically in full coarsening along a particular *tie-shaped central* portion of the grid and in semi coarsening as the proximity to the domain edges grow. All coarser grids add a band of (full-coarsened) cells to the right and the left of this tie-shaped portion, adding up to complexity reduction. Figure 4 gives the complete sequence of coarse grids generated for a 64^2 fine grid stretched with $\alpha = 1.03$. In Section 5, we solve the model problem on this sequence of grids and demonstrate the convergence of the resulting multigrid solver.

Remark 1. (Aspect-ratio bound for good multigrid convergence) *We would like to highlight that the aspect ratio of a particular cell is tolerable up to a value of 1.3 from the point of view of multigrid convergence [14]. Seeking the value 1.0 for prospective agglomerations is idealistic and impractical, often leading to coarsening choices that imply very poor reduction of unknowns per grid level. This observation leads us to use an SQTOL value of 0.3 in our numerical experiments.*

Remark 2. (Nodal position on the coarened grid) *After the coarsening process is over and the coarse grid is constructed, we place the nodes (i.e., unknowns) in the center of the coarse-grid control volumes.*

3.2. The DCG operator and point-based relaxation

3.2.1. The DCG operator On the coarse grids, we use a matrix-free discretization coarse-grid operator. Another choice (matrix based), would be the *GCG (the Galerkin Coarse-Grid operator)*. As the name implies, the DCG operator is obtained by direct discretization on the coarse grids. The discretization scheme on the coarse grids is the same as that on the finest grid, i.e. the cell-centered FVM. Consider Figure 3(b);



(a) Level 1 (b) Level 2 (c) Level 3

(d) Level 4 (e) Level 5 (f) Level 6

Figure 4. The complete sequence of coarse grids obtained by coarsening a 64^2 grid stretched with $\alpha = 1.03$, by the described algorithm

if the m^{th} control volume is defined by the rectangle $\overline{ABCD}$, then;

$$\int_A^B a_2 \frac{\partial u_m}{\partial y} dx_m - \int_B^C a_1 \frac{\partial u_m}{\partial x} dy_m - \int_C^D a_2 \frac{\partial u_m}{\partial y} dx_m + \int_D^A a_1 \frac{\partial u_m}{\partial x} dy_m = \int_{\Omega_m} f^{\Omega} d\Omega_m \quad (5)$$

follows directly from (3). The right-hand side of (5) can be approximated simply by the point value of the source function multiplied by the volume of the cell. Each first-order derivative on the left-hand side is approximated by the central $O(h^2)$ FDM if the particular k^{th} face of the control-volume boundary is not a portion of the domain boundary. However, if it is, then the derivative is approximated by the $O(h)$ one-sided FDM. For a uniform equidistant-grid layout, this scheme results in second-order accuracy [21].

For the finest grid, it is important to point out that the discretization of the derivatives in Eqn. (5) -being through finite differences- is trivial due to the perfect alignment of the nodes with their horizontal and vertical neighbours. However, on coarser grids, this becomes more involved due to the presence of hanging nodes.

Example 3. (Coarse-grid stencil for the hanging nodes) *In Figure 3(b), Nodes 2 and 3 do not have horizontal neighbours and, therefore, make use of ghost points g_1 and g_2, respectively, which, in turn, are linearly interpolated from the points directly above and beneath them. In effect, this means that (on the east side) Node 2 is connected with Nodes 11 and 12, while Node 3 is connected with Nodes 12 and 13. The same applies to Nodes 7 and 8 in the horizontal segment. As a general rule, whenever a node is missing, we linearly interpolate it from its collinear neighbours.*

The only general guiding principle for constructing a successful DCG operator is the conservation of flux through each face of the control volume. We define the flux as the *net flow* through a particular face of the cell; for example, the flux through

the east face of the m^{th} control volume is:

$$F_m^{east} = \int_B^C a_1 \frac{\partial u_m}{\partial x} dy_m$$

Remark 3. (Alternative flux definition) *In contrast to our definition of the flux as the net flow through a boundary, flux is also often defined as the rate of flow, i.e., without incorporating the length of the boundary segment (dy_m in this case) in the definition. This is useful where flux has to be averaged across control volumes such as might be encountered in an AMR setting [1]. In our work, however, defining flux as the net flow is more helpful. Mathematically the two definitions are equally acceptable.*

Example 4. (Conservation of flux) *For Nodes 2, 3, and 12 of Figure 3(b), conservation of flux would mean that the following equality holds:*

$$F_{12}^{west} = -\left(F_2^{east} + F_3^{east}\right)$$

For the purpose of illustration and completeness, we derive the discrete flux balance equations for Nodes 2 *and* 12 *in the Appendix. The same method carries over to all control volumes found in any of the coarse grids, with or without hanging nodes in their proximity.*

3.2.2. Point-based relaxation The relaxation process in our multigrid method is a variant of the lexicographical point-based Gauss-Seidel. The variation is only in terms of the pattern in which the domain is traversed. The smoothing properties of a stationary iterative method, such as Gauss-Seidel, are not invariant of the relaxation pattern in which the unknowns are visited. The traversal pattern in this work, trivially, is in L-shaped strips following the enumeration of the domain in these structures, and the ordering of the control volumes within them. Each L-strip is visited in the enumeration order. Within each strip, first the node in the vertex control volume is relaxed, then each of the nodes (in enumeration order) in the control volumes in the vertical segment, and finally the nodes in the horizontal segment of an L-strip. The observed smoothing properties are superior to those of classical lexicographic Gauss-Seidel and slightly inferior to Red-Black Gauss Seidel; however, they are sufficiently good to provide excellent multigrid convergence in this set-up.

3.3. The transfer operators

3.3.1. The restriction operator We use piece-wise constant restriction to transfer grid functions from finer grid levels to coarser grid levels. In our cell-centered setting, due to variable mesh sizes, this requires careful averaging across cells that are greatly different in volume. Therefore, we use a volume-average based restriction.

Here, we explain the method through which fine-grid cells are restricted to coarse-grid cells. We denote by m the index of a subset of control volumes of the finer level, which would be agglomerated to form the control volume, M, after the coarsening decision has taken place.

Nodes m of fine level l contain values averaged over their respective control volumes, i.e.:

$$u_m = \frac{\int_{\Omega_m} u \, d\Omega_m}{|\Omega_m|}, \tag{6}$$

and, similarly, Node, M, should contain a value representing the average over the control volume M of coarse level $(l+1)$,

$$u_M = \frac{\int_{\Omega_M} u \, d\Omega_M}{|\Omega_M|} \tag{7}$$

which gives:

$$\begin{aligned} u_M &= \left(\sum_{\Omega_m \in \Omega_M} \int_{\Omega_m} u \, d\Omega_m \right) / \, |\Omega_M| \\ &= \left(\sum_{m \in M} u_m \, |\Omega_m| \right) / \, |\Omega_M| \end{aligned}$$

This averaging formula contains a contribution from each of the fine-grid cells in this subset, respective to their volumes. This cumulative contribution is then distributed over the coarse-grid cell volume, to represent an average value within it. All fine to coarse transfers in our work make use of this restriction.

3.3.2. The prolongation operator We use simple node-position based bilinear interpolation for transfering grid functions from coarse to fine levels. From a global perspective, the nodes are ordered into horizontal and vertical segments of L-strips. Each fine-grid horizontal or vertical segment has a similarly oriented coarse-grid segment to its left and right. In general, the nodes on these left and right neighbours are not aligned with the fine-grid nodes and, therefore, first have to produce linearly interpolated values which are collinear with the fine-grid nodes. After these values have been interpolated in one dimension, the fine-grid nodes are subsequently interpolated from them. The interpolation (analogous to restriction) is based on the actual relative distancing of the nodes, and not on the fixed component prolongation stencils. This process ensures that each fine-grid node is interpolated with 4 coarse nodes (from all four surrounding sides), and that the interpolation process is never uni-directional.

Example 5. (Interpolating a fine-grid L-strip) *Consider Figure 5. Grid-levels* 3 *and* 4 *of a particular grid sequence are superimposed to elaborate on how bilinear interpolation takes place in the L-shaped setting. The coarse grid is represented by thick grey lines, and the fine-grid by fine black lines. L_1 and L_2 strips of the coarse-grid are shown shaded along with their nodes, represented by hollow grey circles. These coarse-grid L-strips enclose L_3 and L_4 strips of of the fine-grid, containing solid black and solid white -shaded nodes, respectively. In the figure, the interpolation for the black nodes is demonstrated; white nodes are only there to emphasize that they too would have to be interpolated from the same set of coarse nodes. The black crosses represent ghost positions, collinear both with the coarse-grid nodes and (black) fine-grid nodes. The interpolation takes place in two stages. Proceeding from (c), first the crosses are interpolated linearly from the coarse-grid nodes (hollow grey circles). Once these ghost points are populated, the original coarse-grid nodes have no role left. They are deliberately not shown in (e) to depict this. At this stage, the black fine-grid nodes are finally interpolated linearly from the crosses with which they are collinear. This scheme gives bilinear accuracy. Note that interpolation of the solid white fine-grid nodes would employ the same coarse-grid nodes, but the position of the crosses would change and reflect collinearlity with the white nodes.*

The interpolation coefficients are only computed for the vertical segment and are retained for use with the horizontal segment to take advantage of symmetry. Each L-strip (and, subsequently, each control volume in it) is treated in the enumeration order. The nodal values adjacent to the domain boundary are interpolated from coarse-grid values on one side and from the boundary value on the other side. This scheme turns out to be a better transfer than interpolating the boundary nodes linearly from only one side.

Remark 4. (Final assembly into the multigrid algorithm) *The actual solution process has a small setup phase in which all of the coarse grids are constructed and stored. The storage complexity for any grid never exceeds the number of control volumes in it. This is*

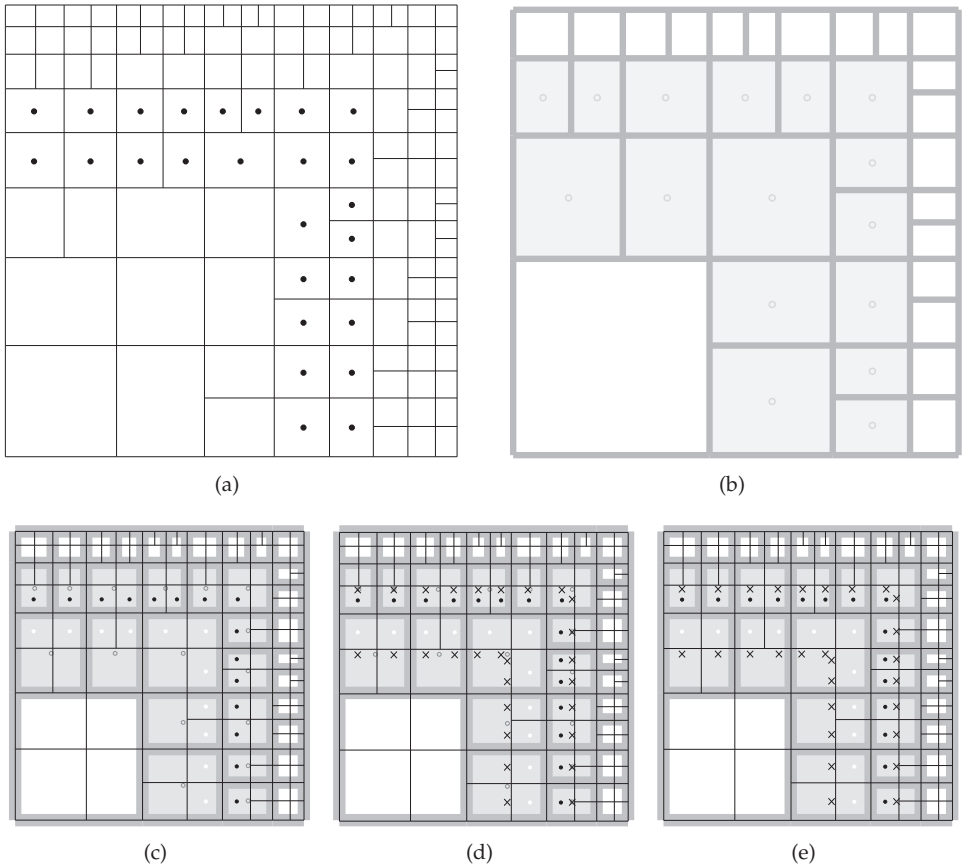


Figure 5. (a) and (b) represent the third and the fourth levels respectively of a 32^2 grid stretched with $\alpha = 1.06$, respectively. In (c), (d), and (e), these levels are superimposed to demonstrate the prolongation for level 3 from level 4. Each fine-grid node is interpolated by information from all 4 surrounding sides.

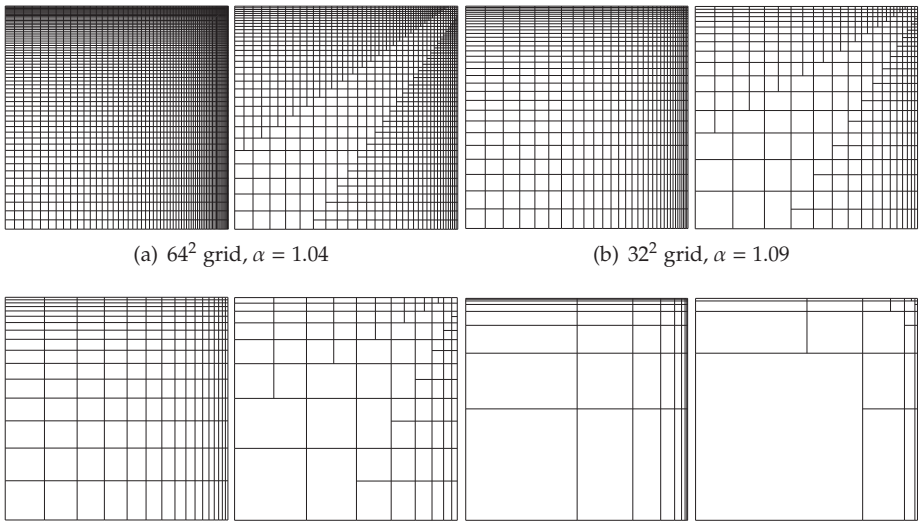
managed through a data structure embodying the L-shape enumerated grid, which yields all grid parameters, including the mesh sizes, the nodal positions, symmetry information, etc., and is, therefore, indexed to point to a particular member within a family of grids. It is important to point out that the natural form of discretization of FVM or FEM (unlike FDM) has the right-hand side scaled. This implies that the residual also has the same scaling, and, therefore, must be neutralized and re-scaled both before and after restriction to the coarse grid.

4. Complexity

We measure the complexity of our multigrid method in terms of the computational work which, at grid level k, is given recursively as:

$$W_k = W_k^{k+1} + \gamma_{k+1} W_{k+1}; \qquad k = 0, 1, 2, \cdots, l-1$$

where γ_i is the cycle index to be used on the i^{th} level, and W_k^{k+1} is the amount of work required during a 2-grid cycle involving the k^{th} and the $(k+1)^{th}$ grid levels [1]. W_k^{k+1} involves pre -and post- smoothing and subsequent grid transfers to and from the k^{th} grid level. For most reasonable choices of the multigrid components,



(a) 64^2 grid, $\alpha = 1.04$ (b) 32^2 grid, $\alpha = 1.09$

(c) 16^2 grid, $\alpha = 1.20$ (d) 8^2 grid, $\alpha = 2.00$

Figure 6. A variety of fine and coarse grids with a reduction factor ≈ 2.34

it is justified to assume that:

$$W_k^{k+1} \leq C\, M_k; \qquad k = 0, 1, 2, \cdots, l-1 \tag{8}$$

where C is a small constant, and M_k represents the number of unknowns that exist on grid level k. Computational work is measured in work-units, wu, which we define as C times the cost of 1 relaxation on the finest grid, Ω_0. i.e.,

$$1\ wu = C\, M_0 \tag{9}$$

Freezing the cycle index to a fixed value, the total work, W is bounded by:

$$W \leqslant C\left[M_0 + \gamma M_1 + \gamma^2 M_2 + \cdots + \gamma^{l-1} M_{l-1}\right]. \tag{10}$$

Observing the method from a 2-grid perspective helps in estimating its complexity. In Figure 6, the grid reduction in a 2-grid setting is displayed. The specific tie-shapes are the regions where 4-cell agglomerations take place, whereas in the darker regions close to the domain edges, semicoarsening in either direction is performed. It has already been discussed that this hybrid behaviour pays off in the form of better complexity values. In this particular figure, the grid sizes range from 8^2 to 256^2, and α ranges from 1.01 to 2.0. Although these grids appear greatly different in their layout, they have a common denominator, as they share a common measure of the worst aspect ratio (which has been deliberately brought about by specific combinations of grid sizes and stretching parameters). In turn, all of them display a grid reduction factor around 2.34. This directly suggests that the coarsening factor depends greatly on the maximal aspect ratio. For a closed form estimate of the complexity in a 2-grid setting, we consider the first two grids. The control volumes on the fine grid are aligned both horizontally and vertically and, therefore, the cell index i runs from 0 to $(\sqrt{M_0} - 1)$ in both directions. The mesh-sizes are governed by Equation (4), and after a slight modification, are given by;

$$h_i = \frac{x_{max}}{\left(\frac{\alpha^{M_0}-1}{\alpha-1}\right)}\alpha^i, \qquad i = 0, \cdots, n, \cdots, (M_0 - 1)$$

	α					
M_0	1.00	1.02	1.04	1.06	1.08	1.10
4^2	4	4	4	4	4	4
8^2	4	4	4	4	3.28	3.29
16^2	4	4	3.50	3.51	3.58	3.23
32^2	4	3.62	3.40	3.22	2.96	2.87
64^2	4	3.52	3.17	2.72	2.64	2.50
128^2	4	3.22	2.72	2.45	2.36	2.34
256^2	4	2.77	2.37	2.29	2.27	2.26

Table I. A closed form for $\tau(M_0, \alpha)$ is not readily available and, therefore, discrete (empirically observed) values -averaged over a sequence of l grids- are shown for a diverse combination of grid sizes and α

During the inspection of the vertical segment for coarsening decisions (as outlined in Section 3.1), the index of the last cell (on the first strip L_0) chosen for full (2×2) coarsening, is represented by n. Once n is picked, it is fairly straight forward to connect the first two grids in a closed form formula.

Let c_1 denote the number of cells in the coarse grid that were obtained by agglomerating 4 cells (2×2 coarsening) of the fine grid, and c_2 denote the remaining coarse-grid cells, obtained from semi-coarsening. c_1 depends on the value of n picked by the coarsening algorithm, which always reports the form of c_1 given in Equation (11). The connection between the first two grids is given as;

$$\begin{aligned} c_1 &= \left\{(M_0 - n + 1)n + \sum_{i=1}^{(n-1)} (n - i)\right\} \\ c_2 &= 2(M_0 - 4c_1) \\ M_0 &= (4c_1 + 2c_2)M_1 \end{aligned} \tag{11}$$

For estimating the computational complexity of a multigrid method with a sequence of l grids, i.e., $(\Omega_0, \Omega_1, \cdots, \Omega_{l-1})$, all grids in the sequence have to be taken into account. However, in this situation, a grid-reduction function, $\tau(M_0, \alpha)$, in closed form is not apparent. The relation between the fine-grid level, k, and the coarse-grid level, $(k + 1)$, is as follows:

$$M_k = \tau(M_0, \alpha)\, M_{k+1}, \qquad k = 0, 1, 2 \cdots, (l - 1) \tag{12}$$

Thus Equation (12) and (10) yield:

$$W \leq \frac{\tau(M_0, \alpha)}{\tau(M_0, \alpha) - \gamma}\, C\, M_0 \tag{13}$$

which can be used to evaluate the required work units for a given discrete problem.

Values of $\tau(M_0, \alpha)$ averaged over l grids, for varying combinations of grid sizes and stretching parameters are shown in Table I.

Remark 5. (Compromise between efficiency and grid-reduction) *An efficient multigrid method is always a compromise between grid reduction and the number of iterations that the method takes to converge. Increasing grid reduction would imply cheaper cycles, but higher multigrid convergence factors implying -in turn- a large number of iterations. Decreasing grid reduction implies just the opposite, i.e., relatively lesser number of cycles with each cycle relatively more expensive. The art of designing a good multigrid method depends a great deal on successfully choosing a good compromising strategy.*

For a geometric multigrid method (with standard coarsening) on an equidistant 2d grid, the computational work-per-cycle is bounded by $\frac{4}{3}$. In contrast, a method that only employs semi-coarsening throughout the grid will yield work-per-cycle

	α					
M_0	1.00	1.02	1.04	1.06	1.08	1.10
4^2	1.33	1.33	1.33	1.33	1.33	1.33
8^2	1.33	1.33	1.33	1.33	1.44	1.44
16^2	1.33	1.33	1.40	1.40	1.39	1.45
32^2	1.33	1.38	1.42	1.45	1.51	1.53
64^2	1.33	1.40	1.46	1.58	1.61	1.67
128^2	1.33	1.45	1.58	1.69	1.73	1.75
256^2	1.33	1.56	1.73	1.77	1.79	1.79

Table II. Work estimates, W, computed from (13), and measured in work units, wu.

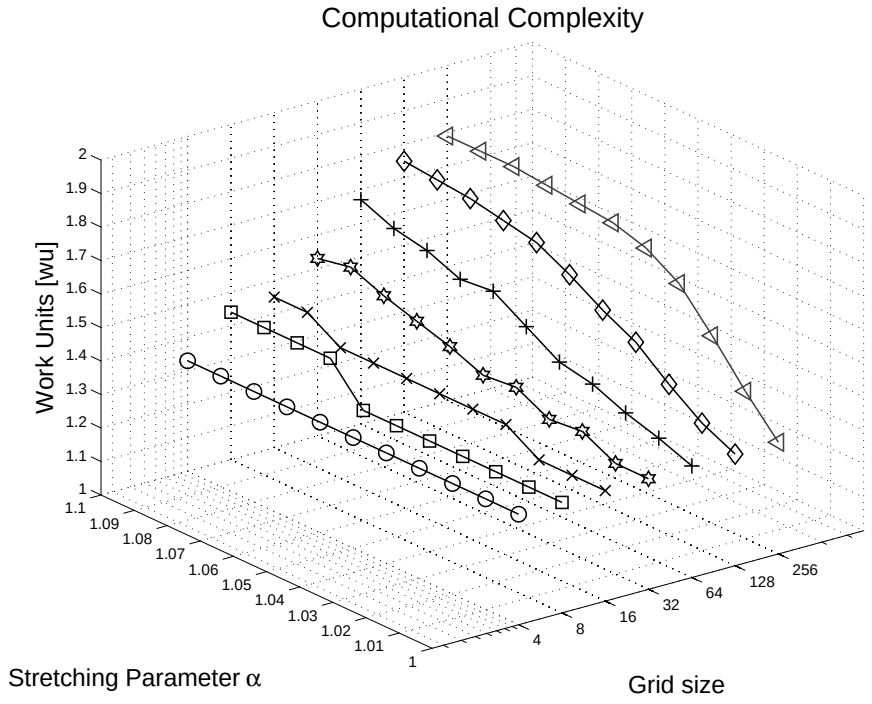
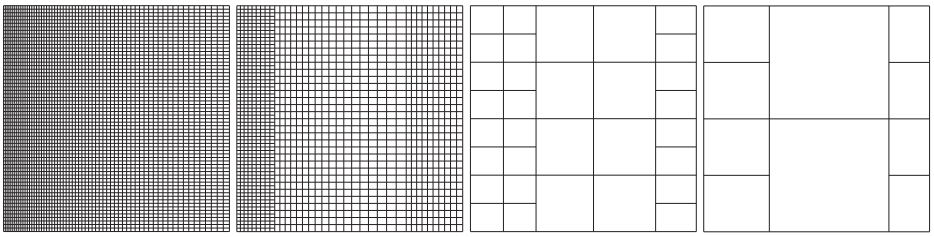


Figure 7. A 3d representation of the complexity of the presented method against the grid size and the stretching parameter.

equal to 2. It is then natural to expect that the method presented here lies between these bounds, as parts of the grid undergo 2×2 agglomeration while other parts only 2×1. In Table II and Figure 7, we present the work estimates for V-cycles ($\gamma = 1$) obtained from substituting the averaged values of the grid-reduction factor τ in (13). The unit of measurement for these estimates is a work unit, wu, as defined in (9). Work-units for grid sizes ranging from 8^2 to 256^2 are shown against values of the stretching parameter α ranging from 1.0 (i.e. no stretching) to 1.1. It is well known that with semi-coarsening-only multigrid algorithms, W-cycle methods do not yield an optimal complexity [1]; however, we see that although our method gets relatively expensive with severe stretching (and consequently bad mesh aspect ratios), the average value of the grid-reduction factor, $\tau(M_0, \alpha)$, always stays above 2.00. This is due to the fact that a semi-coarsening-only strategy is never employed when a grid is being reduced. The computational complexity grows directly with the mesh aspect ratio, which -in our case- is directly related to α as well as the



(a) Level 1 (b) Level 2 (c) Level 5 (d) Level 6

Figure 8. The first two and the last two grids of the sequence for a 64^2 grid stretched only along the x-axis with $\alpha = 1.01$

grid size. Quite apparently for a 256^2 grid, stretching with $\alpha = 1.01$ and $\alpha = 1.02$ is quite sufficient for local grid refinements in practical applications. The complexity in this situation is around 1.5, whereas for an unrealistically high stretching with $\alpha = 1.1$ on 256^2 grid, the complexity is 1.79, still well under the bound set by semicoarsening-only algorithms.

5. Numerical experiments and results

In this section, we demonstrate the presented method at work, by solving boundary value problems based on the model problem described in Section 2. Through the PDE, we approximate the following test function:

$$u(x,\ y) = \frac{sin(2\pi^2 x) + sin(2\pi^2 y)}{(2\pi + x + y)} \tag{14}$$

The results include the residual decay and the achieved multigrid convergence factors. These quantities are displayed against the number of multigrid V-cycles required for convergence. The stopping criterion for convergence is defined by the relative residual going below the tolerance value set at 10^{-8}.

Experiments with 2 kinds of grid stretching are performed. *1-dimensional stretching*, i.e., grids stretched only along the x-axis and equidistant along the y-axis; and *2-dimensional stretching*, i.e., grids stretched along both the axes with the same stretching parameter, α. The multigrid convergence factor is measured empirically by making use of the defect vectors, d_h^k. If an experiment takes m V-cycles to converge, we measure it's quality by the number, $\hat{q}^{(m)}$, which depicts the average defect reduction for m cycles;

$$\hat{q}^{(m)} = \sqrt[(m)]{\frac{\| d_h^m \|}{\| d_h^0 \|}} \tag{15}$$

5.1. Experiments with 1-dimensional grid stretching

The results of 1-dimensional stretching are presented first. The coarsening technique, as described earlier for the 2-dimenional grid stretching, virtually isolates two vertical strips and makes decisions about the different prospective agglomerations. Figure 8 gives the complete grid sequence for an example problem with a 64^2 grid and $\alpha = 1.01$. The guiding principle for building the coarse grids is the same as described in Section 3.1; a prospective agglomeration should *relax* the *tense* aspect ratios. The only notable difference (with the 2-dimensional grid stretching case) is that the control-volume strips are vertical and not L-shaped.

$\alpha_x \rightarrow$ Grid $\downarrow$	1.00	1.02	1.04	1.06	1.08	1.10
16^2	0.13 / 7	0.13 / 7	0.14 / 7	0.15 / 7	0.13 / 7	0.13 / 7
32^2	0.12 / 7	0.15 / 8	0.14 / 7	0.13 / 7	0.13 / 7	0.11 / 7
64^2	0.12 / 7	0.15 / 8	0.12 / 7	0.12 / 7	0.09 / 6	0.08 / 6
128^2	0.11 / 7	0.13 / 7	0.09 / 6	0.09 / 6	0.08 / 6	0.09 / 6

Table III. Multigrid convergence factors against the number of V-cycles (separated by obliques) for different values of α. Grid stretching in these experiments was only along the x-axis.

$\alpha \rightarrow$ Grid $\downarrow$	1.02	1.04	1.06	1.08	1.10
8^2	0.13 / 6	0.13 / 7	0.14 / 7	0.14 / 7	0.14 / 7
16^2	0.17 / 8	0.17 / 8	0.17 / 7	0.15 / 7	0.15 / 7
32^2	0.18 / 8	0.16 / 7	0.16 / 7	0.15 / 7	0.14 / 7
64^2	0.15 / 7	0.14 / 7	0.12 / 6	0.11 / 6	0.11 / 6

Table IV. Multigrid convergence factors and the number of V-cycles (separated by obliques) for different values of α. These experiments are based on 2-dimensional grid stretching

The rest of the process is similar. Multigrid convergence factors for a variety of stretching parameters, α, are displayed in Table III.

We highlight that certain numbers look better than others even for higher values of α. For example, in Table III, the multigrid convergence factor (for the 64^2 grid) for $\alpha = 1.02$ is around 0.15, while for $\alpha = 1.10$, it is around 0.08. This is not an anomaly, -see Remark 5- rather, it points to the fact that the grid reduction for the $\alpha = 1.10$ has been slightly poor in comparison with $\alpha = 1.02$, which implies that more coarse-grid values were available for the interpolation of fine-grid values (on average) and, therefore, the multigrid convergence factors appear somewhat superior at the expense of complexity.

5.2. Experiments with 2-dimensional grid stretching

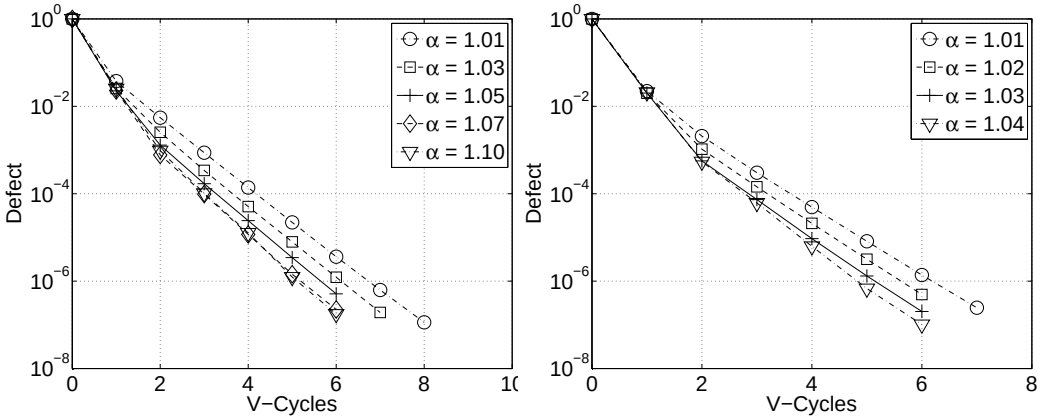
5.2.1. The Model problem The results of experiments with 2-dimensional grid stretching are shown in Table IV. The multigrid convergence factors are quite satisfactory for different α-values giving a range of moderate to severe stretchings. The residual decay for 2-dimensional grid-stretching experiments are presented in Figure 9. Besides good convergence, all of the experiments display a good linear reduction of the residual (measured in the discrete $L2$-norm), around an order of magnitude per V-cycle. This suggests that a full-multigrid algorithm starting on the coarsest grid will converge to the discretization accuracy in just 1 or 2 cycles.

The convergence history for grid-sizes 128^2 and 256^2 (for 2-dimensional stretching) are presented separately in Table V. As these grids are sufficiently fine, they do not allow higher values of α, because it becomes difficult to represent the operator in double precision. The experiments in Table V were performed in 128-bit storage types, to control the round-off errors dominating the computations.

Remark 6. (Stress test) *Figure 10 shows a kind of stress-test. A* 32^2 *grid is stretched with* $\alpha = 5.0$*, and the model problem is solved on this grid with the presented method for the purpose of checking if the method withstands such severe grid stretching. The residual decay is, however, exceptional, around* 20 *orders of magnitude in* 25 *iterations, confirming that the method does not break down. 128-bit floating-point storage and an artificially low*

$\alpha \rightarrow$ Grid $\downarrow$	1.01	1.02	1.03	1.04
128^2	0.15 / 7	0.13 / 6	0.11 / 6	0.10 / 6
256^2	0.12 / 6	0.09 / 5		

Table V. Multigrid convergence factors for 128^2 and 256^2 grids, displayed against the number of V-cycles (separated by obliques) for different values of α. These 2-dimensional grid stretching experiments are presented separately because the discretization scheme breaks down for higher values of α for these larger grids.



(a) 64^2 grid (b) 128^2 grid

Figure 9. Residual decay for 2-dimensional stretching, against the number of V cycles employed.

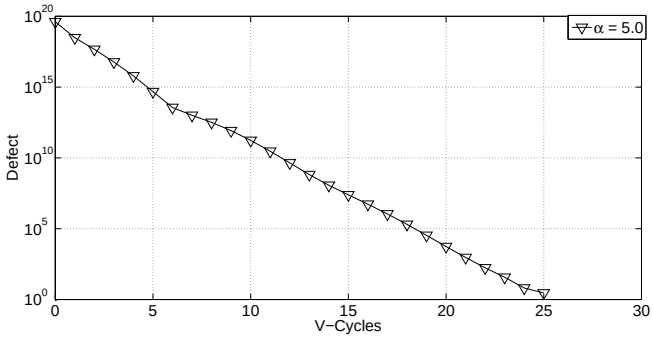
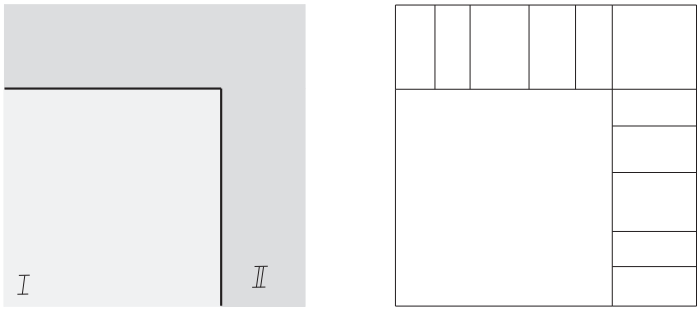


Figure 10. Residual decay for 2-dimensional stretching, against the number of V cycles employed. Note the particularly high stretching for the 32^2 grid

tolerance (equal to machine precision) were used in this experiment.

5.2.2. The model problem with a simple jump discontinuity In this part, we perturb the model problem slightly, by dividing the domain into two parts, as shown in Figure 11(a). The domain is partitioned into an L-shaped subdomain, and the remaining square portion, as demarked by the interface line (solid L-line in Figure 11(a)); so that the Poisson-type equation has different constant coefficients in the different



(a) Domain divided into a square, and an L-shaped subdomains

(b) The coarsest grid L-strip is constructed with the interface line

Figure 11. The subdomains used in the test with a jump discontinuity

$d_1/d_2 \rightarrow$ Grid / α $\downarrow$	2	10^1	10^2	10^3	10^4	10^5
64^2 / 1.03	0.16 / 7	0.33 / 13	0.40 / 16	0.40 / 16	0.40 / 16	0.40 / 16
256^2 / 1.009	0.15 / 7	0.30 / 12	0.35 / 14	0.29 / 12	0.29 / 12	0.29 / 12

Table VI. Multigrid convergence factor for 64^2 and 128^2 problem with jump discontinuity across subdomain interface

parts of the domain but the same analytic solution everywhere. From Equation (1);

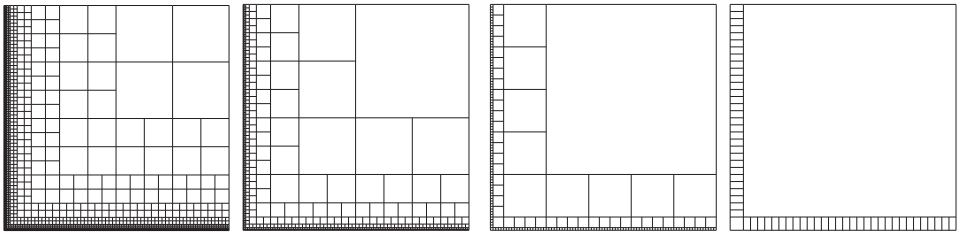
$$a_1 = a_2 = d_1, \qquad (x, y) \in \text{Region I}$$
$$a_1 = a_2 = d_2, \qquad (x, y) \in \text{Region II}$$

The interface line is used in forming the coarsest grid L-strip, which thereby ensures that the interface line never gets annihilated during the construction of coarse grids. As a consequence, prolongation and restriction across this bounding line (discontinuity) takes place at each grid level, moreover, we stay with the prolongation and restriction described in Section 3.3 and do not use operator-dependent transfer operators; neither do we need the Galerkin coarse-grid operator. Table VI shows the $V(1,1)$-cycle results of the presented method when tested with this problem.

We see that the method performs fairly well even with severe jumps across the interface line.

5.2.3. The model problem on amr-type grids Here, we experiment with the model problem on some *amr-type* grids. We show that the methods developed in this paper for *str-type* grids work nicely for certain variants of the *amr-type* grids as well. In these *amr-type* grids, there are no perturbed mesh aspect ratios to improve, and, therefore, the coarsening is always standard, implying a grid reduction factor of at least 4 on all levels. In the present work, we do not perform any implementational adjustments, and simply run the problem on a different grid-sequence. The enumeration is L-shaped and no multigrid components have been altered. The convergence pattern is almost identical to that of *str-type* grids.

The shape of the *amr-type* grids that we use depends on 3 parameters. The number of layers (of different-sized control volumes), the number of control volumes tiled between adjacent layers, and the proportion of cell-volume between cells of adjacent layers; denoted respectively, by n, c, and p. Thus the representation,



(a) Level 1 (b) Level 2 (c) Level 3 (d) Level 4

Figure 12. The grid sequence for the *amr-type* grid, $(8 \times 2,\ 2)$. The finest grid has 8 layers, with 2 cells tiled in each. Cells in adjacent layers are at a volume proportion of 4.

$(n \times c,\ p) \rightarrow$	$(4 \times 2,\ 2)$	$(8 \times 2,\ 2)$	$(8 \times 4,\ 2)$	$(4 \times 32,\ 2)$	$(2 \times 64,\ 2)$
$\hat{q}^{(m)}\ /\ m \rightarrow$	0.16 / 7	0.16 / 7	0.16 / 7	0.16 / 7	0.16 / 7

Table VII. Multigrid convergence factors for different *amr-type* grids.

$(n \times c,\ p)$, describes a particular *amr-type* grid completely. In Figure 12, the complete grid sequence is shown for the finest grid represented by $(8 \times 2,\ 2)$. The coarse grids look quite similar to the ones in Figure 4, the only difference being that, except on the coarsest grid, all the grids in the *amr-type* sequence have the same cell volume within each L-strip. The results of the experiments on these grids are displayed in Table VII, and appear quite satisfactory and stable.

6. Conclusions

A geometric multigrid method based on L-shaped coarsening has been developed and presented in this paper. The method shows near-optimal multigrid convergence with standard components, such as piece-wise constant restriction, simple bi-linear interpolation and a stationary Gauss-Seidel point-based relaxation scheme. The numerical experiments demonstrate that the methods works fine for problems on 2d *str-type* grids, and strongly suggest that the method can be successfully developed to incorporate *AMR* on structured grids. The complexity of the method is comparable with well known approaches like *FAC* and *MLAT*; the main difference with these approaches is that we coarsen the mesh in all regions of the computational domain simultaneously, and this leads to a better complexity measure. A smooth 3d extension of the presented work would imply coarsening decisions proceeding along 3d L-shaped geometric structures. Each of these structures would be composed of 3 half (or partial) planes parallel to the coordinate planes. Similarly to the 2d case, comparisons for coarsening would be between 2d partial coarsening and 3d full coarsening for prospective agglomerations. The results are expected to possess the usual scalability for dimensional extensions of the Poisson equation on equidistant grids.

APPENDIX

Derivation of the discrete equations for Nodes 2 and 12 of Figure 3

To illustrate completely how the discrete coarse grid operator is built, we derive, in this appendix, the discrete flux-balance equations for Nodes 2 and 12 of Figure

3. In the following, k indicates a face of the control volume; specifically, the values $0, 1, 2, 3$ point to the south, east, north, and west faces, respectively. F_m^k denotes the outward flux through the k^{th} face of the m^{th} control volume. f_m^Ω is the source function, f^Ω, computed at (x_m, y_m), which is the Cartesian position of node m. h_m^x and h_m^y are the width and the height (i.e. mesh-sizes along x and y) of the m^{th} control volume, respectively.

For Node $m = 2$ of Figure 3(b), Equation (5) yields:

$$F_2^0 + F_2^1 + F_2^2 + F_2^3 = f_2^\Omega \, (h_2^x \, h_2^y)$$

or

$$-F_1^2 + F_2^1 + F_2^2 + F_2^3 = f_2^\Omega \, (h_2^x \, h_2^y) \tag{16}$$

where

$$\begin{aligned}
F_1^2 &= a_2\Big(x_1,\ y_1 + \tfrac{h_1^y}{2}\Big)\,(u_1 - u_2)\,\Big(\tfrac{2h_1^x}{h_1^y + h_2^y}\Big)\\
F_2^1 &= a_1\Big(x_2 + \tfrac{h_2^x}{2},\ y_2\Big)\,\{u_2 - (c_1 u_{11} + c_2 u_{12})\}\,\Big(\tfrac{2h_2^y}{h_2^x + h_{12}^x}\Big)\\
F_2^2 &= a_2\Big(x_2,\ y_2 + \tfrac{h_2^y}{2}\Big)\,(u_2 - u_3)\,\Big(\tfrac{2h_2^x}{h_2^y + h_3^y}\Big)\\
F_2^3 &= a_1\Big(x_2 - \tfrac{h_2^x}{2},\ y_2\Big)\,\Big\{u_2 - f^\Gamma\Big(x_2 - \tfrac{h_2^x}{2},\ y_2\Big)\Big\}\,\Big(\tfrac{2h_2^y}{h_2^x}\Big)
\end{aligned}$$

Let

$$\begin{aligned}
\beta_1^2 &= a_2\Big(x_1,\ y_1 + \tfrac{h_1^y}{2}\Big)\Big(\tfrac{2h_1^x}{h_1^y + h_2^y}\Big), \qquad \beta_2^1 = a_1\Big(x_2 + \tfrac{h_2^x}{2},\ y_2\Big)\Big(\tfrac{2h_2^y}{h_2^x + h_{12}^x}\Big),\\
\beta_2^2 &= a_2\Big(x_2,\ y_2 + \tfrac{h_2^y}{2}\Big)\Big(\tfrac{2h_2^x}{h_2^y + h_3^y}\Big), \qquad \beta_2^3 = a_1\Big(x_2 - \tfrac{h_2^x}{2},\ y_2\Big)\Big(\tfrac{2h_2^y}{h_2^x}\Big)
\end{aligned}$$

and

$$\beta_2 = \beta_1^2 + \beta_2^1 + \beta_2^2 + \beta_2^3$$

Then, from the flux balance equation (16), we get:

$$\beta_2 \, u_2 = f_2^\Omega \, (h_2^x \, h_2^y) + \beta_1^2 \, u_1 + \beta_2^1 \, (c_1 u_{11} + c_2 u_{12}) + \beta_2^2 \, u_3 + \beta_2^3 \, f^\Gamma\Big(x_2 - \frac{h_2^x}{2},\ y_2\Big) \tag{17}$$

Similarly, for node $m = 12$, we get from (5):

$$F_{12}^0 + F_{12}^1 + F_{12}^2 + F_{12}^3 = f_{12}^\Omega \, (h_{12}^x \, h_{12}^y)$$

or

$$-F_{11}^2 + F_{12}^1 + F_{12}^2 - (F_2^1 + F_3^1) = f_{12}^\Omega \, (h_{12}^x \, h_{12}^y) \tag{18}$$

where

$$\begin{aligned}
F_{11}^2 &= a_2\Big(x_{11},\ y_{11} + \tfrac{h_{11}^y}{2}\Big)\,(u_{11} - u_{12})\,\Big(\tfrac{2h_{11}^x}{h_{11}^y + h_{12}^y}\Big)\\
F_{12}^1 &= a_1\Big(x_{12} + \tfrac{h_{12}^x}{2},\ y_{12}\Big)\,\{u_{12} - u_{18})\}\,\Big(\tfrac{2h_{12}^y}{h_{12}^x + h_{12}^x}\Big)\\
F_{12}^2 &= a_2\Big(x_{12},\ y_{12} + \tfrac{h_{12}^y}{2}\Big)\,(u_{12} - u_{13})\,\Big(\tfrac{2h_{12}^x}{h_{12}^y + h_{13}^y}\Big)\\
F_3^1 &= a_1\Big(x_3 + \tfrac{h_3^x}{2},\ y_3\Big)\,\{u_3 - (c_3 u_{12} + c_4 u_{13})\}\,\Big(\tfrac{2h_3^y}{h_3^x + h_{12}^x}\Big)
\end{aligned}$$

Let

$$\begin{aligned}
\beta_{11}^2 &= a_2\Big(x_{11},\ y_{11} + \tfrac{h_{11}^y}{2}\Big)\Big(\tfrac{2h_{11}^x}{h_{11}^y + h_{12}^y}\Big), \qquad \beta_{12}^1 = a_1\Big(x_{12} + \tfrac{h_{12}^x}{2},\ y_{12}\Big)\Big(\tfrac{2h_{12}^y}{h_{12}^x + h_{18}^x}\Big),\\
\beta_{12}^2 &= a_2\Big(x_{12},\ y_{12} + \tfrac{h_{12}^y}{2}\Big)\Big(\tfrac{2h_{12}^x}{h_{12}^y + h_{13}^y}\Big), \qquad \beta_3^1 = a_1\Big(x_3 + \tfrac{h_3^x}{2},\ y_3\Big)\Big(\tfrac{2h_3^y}{h_3^x + h_{12}^x}\Big)
\end{aligned}$$

and

$$\beta_{12} = \beta_{11}^2 + \beta_{12}^1 + \beta_{12}^2 + c_2\,\beta_2^1 + c_3\,\beta_3^1$$

Then from the flux balance equation (18), we get:

$$\beta_{12}\,u_{12} = f_{12}^{\Omega}\,(h_{12}^x\,h_{12}^y) + \beta_{11}^2\,u_{11} + \beta_{12}^1\,u_{18} + \beta_{12}^2\,u_{13} + \beta_2^1\,c_1\,u_{11} + \beta_3^1\,c_4\,u_{13} \tag{19}$$

Equations (17) and (19) lay emphasis on the way we ensure flux balance in the system. The flux through the south face of Cell 2 is the same quantity computed as Cell 1's north-face flux. Similarly, the flux through the west face of Cell 12 is the cummulative flux flowing in from its west neighbours, i.e. Cells 2 and 3. We reiterate that traversal of the L-strips is in such a fashion that the required fluxes are already available if they have been computed previously. Particularly in connection with this example, note that Cell 1 would be treated earlier than Cell 2, hence its north-face flux is available for use as Cell 2's south-face flux. Similarly Cell 2 and Cell 3 (due to the enumeration scheme) would be treated earlier than Cell 12 and, therefore, their east-face fluxes are already available to add up into the west-face flux of Cell 12.

REFERENCES

1. Trottenberg U, Oosterlee C, Schüller A. *Multigrid*. Academic Press, 2001.
2. Briggs W, Henson V, McCormick S. *A Multigrid Tutorial*. SIAM, 2000.
3. Wesseling P. *An Introduction to Multigrid Methods*. Pure and Applied Mathematics, John Wiley & Sons, 1992.
4. Larsson J, Lien F, Yee E. Conditional semicoarsening multigrid algorithm for the Poisson equation on anisotropic grids. *Journal of Computational Physics* 2005; **208**: 368–383.
5. Morano E, Mavriplis DJ, Venkatakrishnan V. Coarsening strategies for unstructured multigrid techniques with application to anisotropic problems. *Seventh Copper Mountain Conference on Multigrid Methods*, vol. CP 3339, Melson ND, Manteuffel TA, McCormick SF, Douglas CC (eds.), NASA: Hampton, VA, 1996; 591–606.
6. Naik NH, Rosendale JV. The improved robustness of multigrid elliptic solvers based on multiple semicoarsened grids. *SIAM J. Numer. Anal.* 1993; **30**(1):215–229.
7. Mulder WA. A high-resolution Euler solver based on multigrid, semi-coarsening and defect-correction. *Journal of Computational Physics* 1992; **100**: 91–104.
8. Lee B, McCormick SF, Philip B, Quinlan DJ. Asynchronous fast adaptive composite-grid methods: numerical results. *SIAM J. Sci. Comput.* 2003; **25**(2):682–700 (electronic).
9. Berger MJ, Leveque RJ. Adaptive mesh refinement using wave-propagation algorithms for hyperbolic systems. *SIAM J. Numer. Anal.* 1998; **35**(6):2298–2316.
10. Barad M, Colella P. A fourth-order accurate local refinement method for Poisson's equation. *J. Comput. Phys.* 2005; **209**(1):1–18.
11. Martin DF, Colella P, Anghel M, Alexander FJ. Adaptive mesh refinement for multiscale nonequilibrium physics. *Computing in Science and Eng.* 2005; **7**(3):24–31.
12. Haber E, Heldmann S. An octree multigrid method for quasi-static Maxwell's equations with highly discontinuous coefficients. *J. Comput. Phys.* 2007; **223**(2):783–796.
13. Haber E, Heldmann S, Modersitzki J. An octree method for parametric image registration. *SIAM J. Sci. Comput.* 2007; **29**(5):2008–2023.
14. bin Zubair H, Oosterlee C, Wienands R. Multigrid for high-dimensional elliptic partial differential equations on non-equidistant grids. *SIAM J. Sci. Comput.* 2007; **29**(4): 1613–1636.
15. Brandt A, McCormick SF, Ruge JW. Algebraic multigrid (AMG) for sparse matrix equations. *Sparsity and Its Applications*, Evans DJ (ed.). Cambridge University Press: Cambridge, 1984.
16. Ruge JW, Stüben K. Algebraic multigrid (AMG). *Multigrid Methods, Frontiers in Applied Mathematics*, vol. 3, McCormick SF (ed.). SIAM: Philadelphia, 1987; 73–130.
17. Mulder WA. Geophysical modelling of 3d electromagnetic diffusion with multigrid. *Comput Visual Sci* 2008; **11**: 129–138.
18. Rescigno TN, Baertschy M, Isaacs WA, McCurdy CW. Collisional breakup in a quantum system of three charged particles. *Science* 1999; **286**: 2474–2479.
19. Chan T, Xu J, Zikatanov L. An agglomeration multigrid method for unstructured grids. *Contemporary Mathematics, AMS*. 1998; **218**: 67–81.
20. Janka A. Smoothed aggregation multigrid for a Stokes problem. *Computing and Visualization in Science* 2008; **11**:169–180.
21. Wesseling P. *Principles of Computational Fluid Dynamics*. Springer-Verlag New York, Inc.: Secaucus, NJ, USA, 2000.