An Adaptive Multilevel Local Defect Correction Technique with Application to Combustion

Martijn J.H. Anthonissen*, Beth Anne V. Bennett[†], and Mitchell D. Smooke[†]

October 20, 2003

Abstract

The standard local defect correction (LDC) method has been extended to include multilevel adaptive gridding, domain decomposition, and regridding. The domain decomposition algorithm provides a natural route for parallelization by employing many small tensor-product grids, rather than a single large unstructured grid; this algorithm can greatly reduce memory usage. The above properties are illustrated by successfully applying the new algorithm to a simple heat transfer problem with an analytical solution, and by subsequently solving the more complex problem of an axisymmetric laminar Bunsen flame with one-step chemistry. The simulation data show excellent agreement with results previously published in the literature.

1 Introduction

Partial differential equations (PDEs) with solutions that have highly localized properties appear in many application areas, such as combustion, shock hydrodynamics, and transport in porous media. Such problems require a fine grid only in the region(s) of high solution activity, whereas elsewhere a coarser grid suffices. For this reason, the usage of a global uniform fine grid is computationally inefficient; an obvious alternative would be a truly nonuniform refined grid. However, uniform grids retain several advantages over truly nonuniform grids: simple data structures suffice to represent uniform grids; simple, accurate discretization stencils exist for uniform grids; and fast solution techniques are available for solving the system of equations resulting from discretization on uniform grids. For these reasons, so-called local uniform grid refinement techniques have been introduced in which a coarse base grid covering the whole computational domain is locally uniformly refined. These techniques have been applied to elliptic PDEs [21,27,31], hyperbolic PDEs [4,12], and parabolic PDEs [26].

We consider a discretization method for elliptic boundary value problems introduced by Hackbusch [28]. In this technique, called the *local defect correction* (LDC) method, the discretization on the composite grid is based on a combination of standard discretizations on several uniform grids with different spacings that cover different parts of the domain. At least one grid, the coarse grid, must cover the entire domain, and its spacing is chosen in agreement with the relatively smooth

*Eindhoven University of Technology, Department of Mathematics and Computer Science, P.O. Box 513, 5600 MB Eindhoven, The Netherlands, e-mail: m.j.h.anthonissen@tue.nl

[†]Yale University, Department of Mechanical Engineering, P.O. Box 208284, New Haven, CT, 06520-8284, USA

behavior of the solution outside the high activity areas. Apart from this global coarse grid, one or more local fine grids are used that are also uniform, each of which covers only a (small) part of the domain and contains a high activity region. The grid spacings of the local grids are chosen in agreement with the behavior of the continuous solution in that part of the domain. LDC is an iterative process: a basic global discretization is improved by local discretizations defined in subdomains. The update of the coarse grid solution is achieved by adding a defect correction term to the (constant) right-hand side of the coarse grid problem. At each iteration, the process yields a discrete approximation to the continuous solution on the composite grid. The discrete problem that is actually being solved is an implicit result of the iterative process. Therefore, the LDC method is both a discretization method and an iterative solution method.

This paper deals with some extensions to the standard LDC method. We present a technique to discretize and solve elliptic boundary value problems on composite grids generated by adaptive grid refinement. Based on a weight function introduced by Bennett and Smooke [9, 10, 42] that measures the roughness of the solution of the PDE(s) under consideration, high activity areas are flagged for refinement. The combination with LDC, which uses only standard discretizations on rectangular tensor-product grids, is new. The full algorithm is obtained by successively adding adaptivity, multilevel refinement, domain decomposition and regridding to the standard LDC algorithm. After demonstrating the performance of our adaptive multilevel LDC algorithm for a simple problem with an analytical solution, we apply the algorithm to a Bunsen flame problem previously treated by Bennett [42] and Bennett and Smooke [9]. In both references, simulations have been performed with the local rectangular refinement (LRR) solution-adaptive gridding method, in which an unstructured grid is constructed from an initial tensor-product grid by flagging and refining individual grid cells in high activity areas based on a weight function.

The convergence behavior of the LDC algorithm is studied in [1, 2]: an expression for the iteration matrix of the algorithm is derived. For the model problem of Poisson's equation on the unit square with Dirichlet boundary conditions, an upper bound for the (infinity) norm of the iteration matrix is given. An analysis of LDC in combination with finite difference discretizations is presented in [24, 25]. The method is combined with finite volume discretizations in [1, 3]. Nefedov [33] applies LDC in a finite volume context to model fluid flow and heat transfer in a glass tank. LDC is studied in combination with finite element discretizations in [45]. Nefedov and Mattheij [34] consider the LDC method for the case where the global coarse and local fine grids are of a different nature. Whereas most authors consider a composite grid built up of two or more Cartesian grids, Nefedov and Mattheij use a polar grid for the local area.

Other discretization methods on composite grids include the following. McCormick presents the finite volume element (FVE) method, which is used in the fast adaptive composite grid (FAC) method (*cf.* [31, 32, 41]). Ewing, Lazarov and Vassilevski [21, 22] analyze a finite-volume-based local refinement technique with composite grids. Both groups propose an explicit discretization scheme for the composite grid that uses special finite volume stencils near the grid interfaces. The resulting discrete system is then solved by an iterative method (*e.g.* FAC) that may take advantage of the composite grid structure. These special stencils are a crucial difference with the LDC method, which employs only standard discretizations on uniform grids. For the FAC method in a variational setting, convergence results have been given by McCormick [29]. The variational theory is extended to the FVE method in [30] by interpreting FVE as an approximate finite element scheme. Boersma, Kooper, Nieuwstadt and Wesseling [14] present a method for nested-grid calculations in the context of large eddy simulation. As in LDC, their method uses a global coarse grid with one or more local fine grids to cover critical areas. The method employs second-order-accurate finite volume discretizations on the global and local grids, each of which is uniform. Inter-grid communication occurs via artificial Dirichlet boundary conditions. A special interpolation is used for the velocity components normal to the interface to ensure mass conservation.

Adaptive gridding methods have been used by many authors in combustion simulations. Coelho and Pereira [15] model an axisymmetric methane-air diffusion flame using finite volume discretizations on a non-staggered grid. The refinement area is chosen *a priori* by the user, who must also select the maximum refinement level. De Lange and De Goey [18] and Somers and De Goey [40] use a refinement strategy that adds points in areas where a given property has large gradients. Bennett and Smooke successfully apply the above-mentioned LRR method to several axisymmetric combustion problems with both simple and detailed chemical mechanisms [7–11, 38, 42]. The LRR method is compared with the extended LDC algorithm in Section 4.2.3. Becker, Braack and Rannacher [5] develop an adaptive method to simulate two-dimensional flames that uses finite element discretizations rather than finite differences or finite volumes; the latter two methods are most common in numerical combustion. Their adaptive gridding is based on an *a posteriori* error estimator. Finally, Pember *et al.* [35] and Day and Bell [17] investigate axisymmetric laminar premixed and nonpremixed flames, both steady and flickering, using the adaptive mesh refinement (AMR) of Berger and Olinger [13] and Berger and Colella [12]. The AMR technique is contrasted with the present LDC algorithm in more detail in Section 3.2.

2 Formulation of the LDC method

Before presenting our extensions to the LDC method, we begin by describing the standard LDC method and introduce some notation at the same time. We consider the elliptic boundary value problem given by

$$\begin{cases} Lu = f, & \text{in } \Omega, \\ u = g, & \text{on } \partial\Omega. \end{cases} \quad (1)$$

In (1), L is a linear elliptic differential operator, and f and g are the source term and Dirichlet boundary condition, respectively. Other types of boundary conditions can be used as well, but for ease of presentation we formulate the method for (1). To discretize (1), we first choose a global coarse grid (grid spacing H), which we denote by Ω^H . An initial approximation u_0^H on Ω^H can be found by solving the system

$$L^H u_0^H = f^H, \quad (2)$$

which is a discretization of boundary value problem (1). In (2), the right-hand side f^H incorporates the source term f as well as the Dirichlet boundary condition g . We assume L^H to be invertible. We will not be specific about the elliptic operator L and the discrete operator L^H in this section; it may help to think of L as the two-dimensional Laplacian operator and L^H as the standard five-point stencil approximating the Laplacian.

Now, assume that the continuous solution u of (1) has a high activity region in some (small) part of the domain. This high activity of u may be captured by discretizing (1) on a composite grid. To this end, we choose a subdomain $\Omega_l \subset \Omega$ such that the high activity region of u is contained in Ω_l . In Ω_l , we choose a local fine grid (grid spacing h), which we denote by Ω_l^h , such that $\Omega^H \cap \Omega_l \subset \Omega_l^h$; that is, grid points of the global coarse grid that lie in the area of refinement also belong to the local fine grid.

In order to formulate a discrete problem on Ω_l^h , we define artificial boundary conditions on Γ , the interface between Ω_l and $\Omega \setminus \Omega_l$; see Figure 1(a). We apply an interpolation operator $P^{h,H}$ that maps function values at coarse grid points on the interface, denoted by Γ^H , to function values at fine grid points on the interface, denoted by Γ^h . Ferket [23, Sec 4.4] has found that discretization errors usually dominate interpolation errors, regardless of whether linear or quadratic interpolation was applied on the interface. Thus, we take $P^{h,H}$ to be the linear interpolation operator on the interface. In this way, we find the following approximation $u_{l,i}^h$, iteration $i = 0$, on Ω_l^h :

$$L_l^h u_{l,i}^h = f_l^h - B_{l,\Gamma}^h P^{h,H} (u_i^H|_\Gamma). \quad (3)$$

Figure 1: Illustration of the grids used. The darker area is the area of high activity Ω_l . The interface Γ is dashed.

In (3), matrix L_l^h (assumed to be invertible) is a discrete approximation to L on the subdomain Ω_l . The first term on the right-hand side incorporates the source term f as well as the Dirichlet boundary condition g on $\partial\Omega_l \setminus \Gamma$ given in (1). In the second term, the operator $B_{l,\Gamma}^h$ represents the dependence of the fine grid points on the coarse grid solution at the artificial boundary Γ . Once a local solution has been calculated on the finer grid by solving (3) for $u_{l,i}^h$, we can define a composite grid approximation $u_0^{H,h}$ as

$$u_0^{H,h}(x, y) := \begin{cases} u_{l,0}^h(x, y), & (x, y) \in \Omega_l^h, \\ u_0^H(x, y), & (x, y) \in \Omega^H \setminus \Omega_l^h. \end{cases}$$

We will now use the local fine grid solution to update the coarse grid approximation. The grid points of the coarse grid will be partitioned as $\Omega^H = \Omega_l^H \cup \Gamma^H \cup \Omega_c^H$, where $\Omega_l^H := \Omega^H \cap \Omega_l$, $\Gamma^H := \Omega^H \cap \Gamma$ and $\Omega_c^H := \Omega^H \setminus (\Omega_l^H \cup \Gamma^H)$; see Figure 1(b). If we were able to substitute the projection on Ω^H of the exact solution u of boundary value problem (1) into the coarse grid discretization (2), we would find the local discretization error or *local defect* d^H , given by $L^H(u|_{\Omega^H}) = f^H + d^H$. We could then use d^H within the right-hand side of (2) to find a better approximation on the coarse grid. However, as we do not know u , we instead use the fine grid approximation $u_{l,0}^h$ to estimate d^H at the coarse grid points $(x, y) \in \Omega_l^H$. Defining w_0^H as the global coarse grid function of best approximations so far, i.e.

$$w_0^H(x, y) := \begin{cases} u_{l,0}^h(x, y), & (x, y) \in \Omega_l^H, \\ u_0^H(x, y), & (x, y) \in \Gamma^H \cup \Omega_c^H, \end{cases}$$

we then estimate the defect by $d^H = L^H(u|_{\Omega^H}) - f^H \approx L^H w_0^H - f^H =: d_0^H$. Assuming that the stencil at grid point (x, y) involves (at most) function values at $(x+iH, y+jH)$ with $i, j \in \{-1, 0, 1\}$, d_0^H provides an estimate of the local discretization error of the coarse grid discretization at all

points of Ω_1^H . Therefore, we can implement the following *coarse grid correction step* to find u_i^H , $i = 1$, on the coarse grid

$$L^H u_{i+1}^H = f_i^H, \quad (4)$$

where

$$f_i^H(x, y) := \begin{cases} f^H(x, y) + d_i^H(x, y), & (x, y) \in \Omega_1^H, \\ f^H(x, y), & (x, y) \in \Gamma^H \cup \Omega_c^H. \end{cases}$$

Because (4) incorporates estimates of the local discretization error of the coarse grid discretization, the new solution u_1^H is assumed to be more accurate than u_0^H and hence provides a better boundary condition on Γ . A better solution on the local fine grid can be found as before by solving (3) with $i = 1$.

To summarize, we have outlined the following iterative method.

Algorithm 1

Two-grid LDC algorithm with the area of refinement chosen *a priori*

Initialization

- Solve the basic coarse grid problem (2).
- Solve the local fine grid problem (3) with $i = 0$.

Iteration, $i = 1, 2, \dots$

- Solve the updated coarse grid problem (4).
- Solve the local fine grid problem (3).

Often one or two LDC iterations will suffice to obtain a satisfactory approximation on the composite grid due to the high rate of convergence of the method. Typically, iteration errors are reduced by a factor of 10 to 1,000 in each iteration step (cf. the results in [1, 2, 24, 28, 34, 45]).

3 Extensions to the LDC method

We now extend the LDC algorithm by successively adding adaptivity, multilevel refinement, domain decomposition and regridding. The final result will be a technique for discretizing and solving (1) on a composite grid found by adaptive grid refinement, given a code for solving boundary value problem (1) on a tensor-product grid in a rectangular domain.

3.1 Adaptive multilevel refinement

We begin by discussing the first two additions to the LDC algorithm: adaptivity and multilevel refinement. For ease of presentation, we choose the domain Ω to be the unit square, and we assume the initial grid to be uniform; the extension to rectangular domains and tensor-product grids is straightforward. We also assume that the continuous solution u has one area of high activity; it is straightforward to generalize the algorithm to the case where there is more than one area of high activity. We set $x_i := iH$, $y_j := jH$, $i, j = 0, 1, 2, \dots, N$, where the grid spacing H is chosen as $H := 1/N$, with N a positive integer. The initial global coarse grid Ω^H then becomes $\Omega^H := \{(x_i, y_j) \mid i, j = 1, 2, \dots, N-1\}$. We define the boxes B_{ij} formed by grid points and points

on the boundary of the domain such that $B_{ij} = (x_i, x_{i+1}) \times (y_j, y_{j+1})$, $i, j = 0, 1, \dots, N-1$. An initial approximation u_0^H on Ω^H can be found by solving the system (2).

In order to determine which boxes require refinement, we introduce the positive weight function of Bennett and Smooke [9, 10, 42] as an indicator for solution roughness; as detailed below, points will be added in regions where the weight function is large, so it should measure the rapidity of change of u . Defined for each box, the weight function w_{ij} involves first derivatives of the current solution iterate u . We then apply a smoothing filter, an averaging procedure, and a normalization procedure, all of which are described in detail in Bennett and Smooke [9, 10, 42] and will not be repeated here for the sake of brevity. Finally, we determine which boxes need refinement through a subequidistribution procedure [9, 10, 42], in which we flag box B_{ij} for refinement if $w_{ij} > \epsilon$, where the threshold ϵ is a user-specified parameter typically ranging from 1.5 to 3.

We also flag neighboring boxes of high activity boxes for refinement in order to prevent the solution from being artificially trapped at interfaces between coarse and fine grids, which can happen if high activity areas move during recalculation on the finer grid. This latter phenomenon has been observed by Bennett and Smooke [9, 10, 42] in the axisymmetric Bunsen flame, and we will also see it in Section 4.2.3. The number of neighboring boxes of a high activity box that are flagged in each direction is a parameter of the algorithm, which we denote by N_{neighbor} . We find the following collection of all flagged boxes:

$$B_{\text{flagged}} = \{B_{ij} \mid i, j = 0, 1, \dots, N-1, \\ (\exists k, l : k, l = 0, 1, \dots, N-1, \\ i - N_{\text{neighbor}} \leq k \leq i + N_{\text{neighbor}}, \\ j - N_{\text{neighbor}} \leq l \leq j + N_{\text{neighbor}} : w_{kl} > \epsilon)\}. \quad (5)$$

The idea of flagging neighboring boxes is called *layering* in [9, 10, 42]. Note that the thickness of the layer around high activity boxes is expressed in fine grid boxes in [9, 10, 42] rather than in coarse grid boxes as we do in (5). We will elaborate when we formulate the solution procedure.

For the area of refinement Ω_l , we choose the smallest rectangle that encloses all flagged boxes; more efficient choices are discussed later in Section 3.2. In Ω_l , we choose a local fine grid Ω_l^h by uniform refinement of all boxes that constitute Ω_l . The local fine grid has grid spacing $h = H/\sigma$, where the integer refinement factor σ is typically set to 2, although larger values are also allowed. Ideally, σ should be chosen largest at places where the weight function is largest. However, this approach would lead to an unstructured composite grid, which we want to avoid.

Therefore, rather than varying σ within a given run of the code, we will instead use multiple refinement levels. Starting with a coarse base grid, say the Level 0 grid, we can use the procedure sketched above to find a finer Level 1 grid, which itself may be refined using the same procedure recursively to find a yet finer Level 2 grid, etc. In this way, we can keep adding levels of refinement until the finest grid has the desired resolution. Here, we will assume that the maximum level of refinement, denoted by l_{max} , is chosen *a priori*. In practice, we run several simulations, successively adding levels of refinement, in order to choose an appropriate value.

It should be noted that the box-flagging procedure does not need to be identical at all levels. In practice, we set $\epsilon = 2$ at all levels. However, to make sure that the width of the layer is sufficiently large at all levels, the layering parameter should be dependent on the number of the level (l), as in Bennett and Smooke [10, 42]. They choose to surround the originally flagged high activity area with N_{layer} fine grid boxes. Based on a refinement factor $\sigma = 2$, the following recursion for $N_{\text{layer}}(l)$, the layering parameter at Level l , is proposed in [10, form. (6)]:

$$N_{\text{layer}}(l+1) := \max \{2 \cdot (N_{\text{layer}}(l) - 1), 0\} \quad l = 0, 1, 2, \dots \quad (6)$$

Figure 2: Solution procedure for a base grid with two additional levels of refinement.

Note that the minimum in both [10, form. (6)] and [10, form. (7)] is a misprint; it should be replaced with a maximum, as done above in (6). This particular choice (6) for the layering parameter is motivated in [10]. If the layering parameter on the Level 0 grid is chosen larger than 2, the series $N_{\text{layer}}(l)$, $l = 0, 1, 2, \dots$, is increasing. We have expressed the thickness of the layer in coarse grid boxes, as in (5), so we define

$$N_{\text{neighbor}}(l) := N_{\text{layer}}(l) \text{ div } 2, \quad l = 0, 1, 2, \dots \quad (7)$$

in which ‘div’ stands for integer division. Because LDC allows the use of refinement factors larger than 2, we propose the more general expressions

$$N_{\text{layer}}(l+1) := \max \{ \sigma \cdot (N_{\text{layer}}(l) - 1), 0 \}, \quad l = 0, 1, 2, \dots \quad (8)$$

$$N_{\text{neighbor}}(l) := N_{\text{layer}}(l) \text{ div } \sigma, \quad l = 0, 1, 2, \dots \quad (9)$$

After adding successive levels of refinement, the fine grid approximations are used to improve the coarse grid approximations via coarse grid correction steps. Once we have returned to the base grid, we will solve discrete problems on finer levels again. We reuse the same subgrid locations only if the area of interest has not moved; otherwise, new locations should be chosen, as detailed in Section 3.3. At intermediate levels (*i.e.*, neither the coarsest nor the finest level), one LDC iteration step involves two solves: a correction solve when proceeding from the finest level to the coarsest level and a solve with new boundary conditions when proceeding from Level 0 to the maximum level again. During the correction solve, defects are calculated, which are then used in the second solve. The full solution process is shown in Figure 2 for the case of $l_{\text{max}} = 2$.

3.2 Domain decomposition

In the previous section, we determined the smallest rectangle enclosing all flagged boxes and chose to refine this rectangle entirely. However, this approach may refine many boxes that have not been flagged for refinement, especially when an area of high activity is not aligned with the grid directions. To remedy this inefficiency as well as to prevent the grids from becoming too large, we combine the multilevel LDC algorithm with domain decomposition, in which we use a set of rectangles to cover all flagged boxes.

Before describing our algorithm, we mention here that other authors have also developed methods for grouping flagged boxes into rectangles. In the AMR algorithm of Berger and Colella [12],

the refined region consists of a number of rectangular grid patches. Based on a Richardson extrapolation type of estimator for the local discretization error, a sequence of nested logically rectangular grids is created. The authors also use a buffer zone around high activity boxes, and their method to create the rectangular fine grids is heuristic, like ours. Berger and Colella form the smallest rectangle enclosing all flagged boxes and define its efficiency as the ratio of all flagged boxes to the total number of boxes. If the efficiency is less than a preset minimum, the rectangle is split across its long direction, and the process is applied recursively on the two parts. The bisecting step is followed by a merge step: the authors formulate a cost function for the number of operations performed on each grid and decide to merge grids if the single resulting grid has a smaller cost. Berger and Oliger [13] present a similar algorithm but allow rectangles of arbitrary orientation. The AMR algorithm is generalized to three space dimensions in [6].

In the present work, we require each flagged box to be enclosed in at least one rectangle, and we want the rectangles to be overlapping. The overlap of the rectangles is necessary in situations where interfaces between rectangles intersect high activity zones. We remedy large errors at these interfaces by performing a number of domain decomposition iterations via a standard multiplicative Schwarz procedure, in which one alternates between subdomains, solving a new problem each time with boundary conditions updated from the most recent subdomain solutions. For more detail, see [36, Ch. 13], for example.

To decide whether to use more than one enclosing rectangle at a given level, we first determine the smallest rectangle covering all flagged boxes in some coarse grid G and denote this rectangle by $R = (x_{i_1}, x_{i_2}) \times (y_{j_1}, y_{j_2})$. The cost we associate with R is the number of finer grid points found by dividing all coarse grid boxes contained in R with the refinement factor σ . As an alternative, we consider splitting R vertically along index i , $i_1 < i < i_2$, into two smaller rectangles $R_1(i) = (x_{i_1}, x_i) \times (y_{j_1}, y_{j_2})$ and $R_2(i) = (x_i, x_{i_2}) \times (y_{j_1}, y_{j_2})$. The cost of $R_1(i)$, notation $c(R_1(i))$, is the number of finer grid points found by refining the smallest rectangle that encloses all flagged coarse grid boxes in $R_1(i)$ extended with an overlap zone of N_{dd} coarse grid boxes at the right side; N_{dd} is a user-specified integer parameter determining the amount of subgrid overlap. We define the cost of $R_2(i)$ in a similar way but with the overlap added to the rectangle's left side. The cost of vertical splitting can be defined in various ways. A natural choice would be to set

$$c_{\text{vert}}^{(1)} := \min_{i_1 < i < i_2} (c(R_1(i)) + c(R_2(i))), \quad (10)$$

which may give lower costs than using the entire rectangle R , if R contains a large number of unflagged boxes. Note, however, that splitting R has a penalty, because $R_1(i)$ and $R_2(i)$ overlap. An alternative definition of the cost of vertical splitting is

$$c_{\text{vert}}^{(2)} := \min_{i_1 < i < i_2} \max(c(R_1(i)), c(R_2(i))), \quad (11)$$

which induces more splitting to take place, since $c_{\text{vert}}^{(2)} \leq c_{\text{vert}}^{(1)}$. Cost function (11) has the disadvantage that the two finer grids that will be created can differ greatly in size. To remedy this imbalance, we propose the cost function

$$c_{\text{vert}}^{(3)} := \min_{i_1 < i < i_2} (2 \max(c(R_1(i)), c(R_2(i))) - \min(c(R_1(i)), c(R_2(i)))), \quad (12)$$

which we will use in our numerical examples. As there is no particular reason to prefer vertical splitting over horizontal splitting, we can also calculate the cost of splitting R along the index j , $j_1 < j < j_2$. The cost of horizontal splitting is defined in a similar way as the cost of vertical splitting, except that the overlap is now added at the top and bottom of the two smaller subgrids.

Thus far, we have described three options: (i) use the smallest enclosing rectangle R ; (ii) split R vertically along the index which gives the lowest cost and use the smallest enclosing rectangles

Figure 3: Tree representation of a grid hierarchy with two levels of refinement added. The root of the tree corresponds to the base tensor-product grid and is labeled ‘base’. For this particular example, there are three grids at Level 1 and four grids at Level 2.

Figure 4: Solution procedure with domain decomposition for the composite grid of Figure 3.

in R_1 and R_2 ; and (iii) split R horizontally along the index which gives the lowest cost and use the smallest enclosing rectangles in R_1 and R_2 . From among these options, we choose the one with the lowest cost. This choice is overridden (and we refine R entirely) only if the number of points necessary to refine R entirely is less than twice the total number of points of the coarser grid G to which we are applying the domain decomposition algorithm.

If we decide to split R , we repeat the domain decomposition algorithm recursively for the smallest enclosing rectangles in R_1 and R_2 . These recursive calls make yet another subdivision of the grids possible. Hence, the original rectangle R can be split in many smaller rectangles. Each subgrid overlap zone is prohibited from further refinement, in order to avoid refining larger and larger parts of the grids on finer levels, which is both unnecessary and undesirable. The composite grids constructed in the multilevel LDC algorithm extended with domain decomposition may conveniently be represented by a tree, as shown in Figure 3. The algorithm itself, including the Schwarz alternating procedure, is shown in Figure 4 for the composite grid of Figure 3.

Because of the modular design of the code, the solution on each (sub)grid is stored in its own separate file. (A possible drawback of this approach, the relative slowness of communication by files, is discussed in more detail in Section 4.1.) Due to the use of overlap zones, multiple grids at a given level may contain the same grid point. To decide which approximation at that grid point is the most recent (and therefore the most accurate), we track the calculation history by adding an entry to a list each time a calculation takes place on a (sub)grid. Thus, the solution files, together with the list, provide the composite grid approximation.

3.3 Regridding

As mentioned earlier, refining a grid and solving the boundary value problem on the new composite grid may cause the region(s) of high solution activity to move. Areas of the grid may then turn out to have been refined unnecessarily or areas not previously refined may now require refinement. Therefore, we apply the following regridding procedure from [9, 42] before proceeding from Level l to Level $l + 1$. First, we store the current composite grid solution and also project it onto the Level 0 grid. We choose new Level 1 grids by calculating the weight function based on this projected solution and using the domain decomposition algorithm. Next, the stored composite grid solution is projected on the new Level 1 grids; based on these approximations, Level 2 grids are chosen. This procedure continues until we have chosen the Level $l + 1$ grids.

4 Applications

4.1 An example problem

To illustrate the working of the LDC algorithm combined with domain decomposition, we consider the following model problem (the “heated plate” problem) taken from [10, 42]:

$$\begin{cases} -\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + \frac{\partial u}{\partial x} + \frac{\partial u}{\partial y} = f, & \text{in } \Omega = (0, 1) \times (0, 4), \\ u = g, & \text{on } \partial\Omega. \end{cases} \quad (13)$$

In (13), f and g have been chosen such that

$$u(x, y) = \tanh[\alpha(-4x - 2y + 3)] + 1, \quad \alpha = 5. \quad (14)$$

The continuous solution u is approximately 2 in the lower left part of the domain. It makes a sharp jump over the line $4x + 2y = 3$ and is approximately 0 in the rest of the domain.

For the discretization of (13), we begin with a uniform coarse grid Ω^H in Ω with spacings $\Delta x = \Delta y = H = 1/5$, as in [10, 42]. Both the diffusion and convection terms in (13) are discretized using standard centered differences on all grids. We carry out five simulations on a 600 MHz Pentium III with 128 MB of RAM; the difference among these runs is the maximum refinement level $l_{\max}$. We use the following settings: the threshold value ϵ for the weight function is set to 2 at all levels; the layering parameter is chosen according to (6), (7) with $N_{\text{layer}}(0) = 3$; refinement factor $\sigma = 2$; and overlap parameter $N_{\text{dd}} = 1$. At levels with more than one grid, the problems on the subgrids are solved sequentially. Two solves take place on each subgrid before proceeding to the next level. For this problem, grid refinement does not cause the high activity area to move; with or without regridding, the results were identical. Figure 5 shows the composite grid and the error of the LDC solution for the $l_{\max} = 3$ simulation.

For comparison reasons, solutions have been calculated on uniform grids with grid spacings equal to the finest spacings of the LDC composite grids. For these uniform grid simulations, the solver is the same as that used in the LDC subgrid solves (namely, the Newton solver described in Section 4.2.2), and the initial guess is that used by [10, 42] but with a floor of 0 and a ceiling of 2. All numerical results are listed in Table 1. The errors are maximum errors, and the number of LDC iterations is set to 1; additional iterations did not decrease the error of the LDC approximations. We find that the error on the LDC composite grids is practically equal to the error on the equivalent uniform grid for all simulations. Only in the $l_{\max} = 5$ simulation does the error not decrease, which can be explained as follows. In the $l_{\max} = 4$ simulation, the steep solution

Figure 5: The composite grid and the error of the LDC solution for the simulation with $l_{\max} = 3$.

gradient is well resolved, and the maximum error is located at a Level 1 grid point outside the area that would be refined by introducing more levels. For this reason, adding the Level 5 grid does not decrease the (maximum) error on the composite grid.

The entries in the fourth columns of Tables 1(a) and 1(b) are the execution times of the calculations in seconds. We have averaged the solution times of ten identical runs for each simulation in order to minimize timing inaccuracies. The times of Table 1(b) may seem disappointing at first glance, but recall that we have chosen to write separate programs to increase the modularity of the code. These programs communicate via files, which is relatively expensive compared to the solution times for this simple model problem. In addition, rather than solving the problems on nonoverlapping subgrids at a given level in parallel, we have solved them sequentially. On the Level 4 grid, for example, the problems on the 8 subgrids have been solved one after another.

The LDC algorithm with domain decomposition far outperforms the uniform grid solver with respect to memory usage. The solver's memory usage scales approximately linearly with the number of grid points. For the uniform grid having 103,201 points, the corresponding LDC composite grid has a total of 33,767 points comprising 32 subgrids. The largest constituent subgrid has only 1,763 points and is located at Level 5 (see Table 2), which implies that calculations take place on grids with 1,763 points or fewer — a huge memory savings over the single solve on the uniform grid with 103,201 points. In addition, a reduction in the number of points may also lead to faster convergence of the solver used on the individual tensor-product grids.

H	Points	Error	Time
1/5	126	$2.45 \cdot 10^{+0}$	0.07
1/10	451	$3.43 \cdot 10^{-1}$	0.12
1/20	1,701	$9.43 \cdot 10^{-2}$	0.36
1/40	6,601	$1.94 \cdot 10^{-2}$	1.76
1/80	26,001	$4.66 \cdot 10^{-3}$	10.63
1/160	103,201	$3.00 \cdot 10^{-3}$	59.53

(a) Uniform grids.

$l_{\max}$	Grid points	Error	Time
0	126	$2.45 \cdot 10^{+0}$	0.08
1	+231 = 357	$3.43 \cdot 10^{-1}$	0.69
2	+824 = 1,181	$9.42 \cdot 10^{-2}$	2.65
3	+2,480 = 3,661	$1.92 \cdot 10^{-2}$	7.13
4	+7,328 = 10,989	$5.02 \cdot 10^{-3}$	20.14
5	+22,778 = 33,767	$5.02 \cdot 10^{-3}$	54.70

(b) LDC composite grids.

Table 1: Maximum norms of the error and execution times (in seconds) of the numerical simulations for problem (13).

Level	Grid	Grid points	Sum	Total
0	base	126	126	126
1	base.1	231	231	357
2	base.1.1	399	824	1,181
	base.1.2	425		
3	base.1.1.1	693	2,480	3,661
	base.1.1.2	627		
	base.1.2.1	551		
	base.1.2.2	609		
4	base.1.1.1.1	987	7,328	10,989
	base.1.1.1.2	1,127		
	base.1.1.2.1	893		
	base.1.1.2.2	893		
	base.1.2.1.1	891		
	base.1.2.1.2	825		
	base.1.2.2.1	875		
	base.1.2.2.2	837		
5	base.1.1.1.1.1	1,599	22,778	33,767
	base.1.1.1.1.2	1,599		
	base.1.1.1.2.1	1,763		
	base.1.1.1.2.2	1,681		
	base.1.1.2.1.1	1,353		
	base.1.1.2.1.2	1,287		
	base.1.1.2.2.1	1,353		
	base.1.1.2.2.2	1,287		
	base.1.2.1.1.1	1,395		
	base.1.2.1.1.2	1,365		
	base.1.2.1.2.1	1,435		
	base.1.2.1.2.2	1,457		
	base.1.2.2.1.1	1,395		
	base.1.2.2.1.2	1,395		
	base.1.2.2.2.1	1,189		
	base.1.2.2.2.2	1,225		

Table 2: Details of the LDC composite grid for problem (13), for the simulation with $l_{\max} = 5$.

4.2 Bunsen flame

We now turn our attention to a more complex application: the axisymmetric laminar Bunsen flame with one-step chemistry. This problem was previously presented by Bennett and Smooke in [9, 42] as an application of the LRR method. Because almost all of the dependent variables in the Bunsen flame problem have large gradients in a very small region of the computational domain, adaptive gridding is a must for this simulation.

The physical configuration for the Bunsen flame is shown in Figure 6. A mixture of methane and air flows up from a central jet, which is surrounded by a coflowing air stream. A steady conical

Figure 6: Physical configuration for the axisymmetric Bunsen flame.

flame forms at the mouth of the cylindrical burner, and the base of this flame is slightly wider than the diameter of the inner jet. The equivalence ratio Φ of this lean premixed flame is 0.776. As indicated in the figure, the central jet has inner and outer radii r_1 (0.5 cm) and r_2 (0.55 cm), respectively, and the surrounding jet has radius r_3 (3 cm).

The chemical model we consider has 5 species: methane (CH_4); oxygen (O_2); water (H_2O); carbon dioxide (CO_2); and the abundant inert, nitrogen (N_2). The global chemical behavior of the gas mixture is described by

There are 9 dependent variables in the Bunsen flame problem: radial velocity (v_r); axial velocity (v_z); vorticity (ω); temperature (T); and 5 mass fractions (Y_{CH_4} , Y_{O_2} , $Y_{\text{H}_2\text{O}}$, Y_{CO_2} , and Y_{N_2}). These variables satisfy the following strongly coupled, nonlinear PDEs.

$$\frac{\partial^2 v_r}{\partial r^2} + \frac{\partial^2 v_r}{\partial z^2} = \frac{\partial \omega}{\partial z} - \frac{1}{r} \frac{\partial v_r}{\partial r} + \frac{v_r}{r^2} - \frac{\partial}{\partial r} \frac{\mathbf{v} \cdot \nabla \rho}{\rho}, \quad (16)$$

$$\frac{\partial^2 v_z}{\partial r^2} + \frac{\partial^2 v_z}{\partial z^2} = -\frac{\partial \omega}{\partial r} - \frac{1}{r} \frac{\partial v_r}{\partial z} - \frac{\partial}{\partial z} \frac{\mathbf{v} \cdot \nabla \rho}{\rho}, \quad (17)$$

$$\begin{aligned} \frac{\partial^2}{\partial r^2}(\mu\omega) + \frac{\partial^2}{\partial z^2}(\mu\omega) = & -\frac{\partial}{\partial r} \left(\frac{\mu\omega}{r} \right) + \rho v_r \frac{\partial \omega}{\partial r} + \rho v_z \frac{\partial \omega}{\partial z} - \frac{1}{r} \rho v_r \omega + \overline{\nabla} \rho \cdot \nabla \left(\frac{\mathbf{v} \cdot \mathbf{v}}{2} \right) \\ & - \overline{\nabla} \rho \cdot \mathbf{g} + 2 \left\{ (\overline{\nabla}(\nabla \cdot \mathbf{v})) \cdot \nabla \mu - \nabla v_r \cdot \overline{\nabla} \frac{\partial \mu}{\partial r} - \nabla v_z \cdot \overline{\nabla} \frac{\partial \mu}{\partial z} \right\}, \end{aligned} \quad (18)$$

$$c_p \left(\rho v_r \frac{\partial T}{\partial r} + \rho v_z \frac{\partial T}{\partial z} \right) = \frac{1}{r} \frac{\partial}{\partial r} \left(r \lambda \frac{\partial T}{\partial r} \right) + \frac{\partial}{\partial z} \left(\lambda \frac{\partial T}{\partial z} \right) - Q w_{\text{CH}_4}, \quad (19)$$

$$\rho v_r \frac{\partial Y_i}{\partial r} + \rho v_z \frac{\partial Y_i}{\partial z} = \frac{1}{\text{Le}_i} \left\{ \frac{1}{r} \frac{\partial}{\partial r} \left(r \frac{\lambda}{c_p} \frac{\partial Y_i}{\partial r} \right) + \frac{\partial}{\partial z} \left(\frac{\lambda}{c_p} \frac{\partial Y_i}{\partial z} \right) \right\} + w_i. \quad (20)$$

Radial coordinate		Axial coordinate	
Range	Grid spacing	Range	Grid spacing
0–0.76	0.01	0–0.56	0.01
–0.80	0.02	–0.80	0.02
–0.90	0.025	–0.90	0.025
–1.00	0.05	–1.00	0.05
–1.40	0.1	–1.30	0.1
–2.00	0.2	–1.50	0.2
–3.00	0.25	–2.50	0.25
		–4.00	0.5
		–7.00	1
		–25.00	2

Table 3: Radial (r) and axial (z) coordinates (in centimeters) of the initial tensor-product grid for the Bunsen flame problem.

In (16)–(20), we have used the following symbols: the mixture density ρ , determined from the ideal gas law; the Lewis number Le_i of species i ($Le_{CH_4} = 0.96$, $Le_{O_2} = 1.1$, $Le_{H_2O} = 0.83$, $Le_{CO_2} = 1.39$, and $Le_{N_2} = 1$); the thermal conductivity λ , determined from $\rho\lambda = 2.3681 \cdot 10^{-2}$ kg J/(m⁴ s K); the specific heat at constant pressure c_p ($1.2887 \cdot 10^3$ J/(kg K)); the reaction rates w_i ; the viscosity coefficient μ , determined from setting the Prandtl number, *viz.* $Pr := \mu c_p / \lambda$, to 0.7; the gravitational acceleration $\mathbf{g} = -g\mathbf{e}_z$ with $g = 9.82$ m/s²; the differential operator $\bar{\nabla}$, defined as $\bar{\nabla} = (\partial/\partial z, 0, -\partial/\partial r)^T$; and the heat release Q per unit mass of fuel consumed ($4.9823 \cdot 10^7$ J/kg). Note that (20) is applied for only four of the five mass fractions; Y_{N_2} is calculated from the condition that all mass fractions must sum to unity. From equation (15), the reaction rates w_i of the individual species can be related as follows:

$$w_{CH_4} = -W_{CH_4} w, \quad w_{O_2} = -2W_{O_2} w, \quad w_{H_2O} = 2W_{H_2O} w, \quad w_{CO_2} = W_{CO_2} w, \quad (21)$$

with

$$w = A \frac{\rho Y_{CH_4}}{W_{CH_4}} \left(\frac{\rho Y_{O_2}}{W_{O_2}} \right)^2 \exp \left(-\frac{E}{RT} \right). \quad (22)$$

Here, W_i is the molar weight of species i , A is the frequency factor ($8.3456 \cdot 10^8$ m⁶/(mol² s)), E is the activation energy ($1.2594 \cdot 10^5$ J/mol), and R is the universal gas constant (8.3147 J/(mol K)). All of the above-listed parameter values are taken from [9, Table 1], which, in turn, come from [16] for the case of $\Phi = 0.776$.

It should be noted that the Arrhenius reaction rate pre-exponential factor in [9] is different from ours in (22). If we denote the factor from [9] by $\tilde{A}$, then $A = \tilde{A} W_{O_2}^2$. The source terms in the species continuity equations and in the energy equation in [9] are expressed in the auxiliary variable q , which is related to w from (22) via $q := QW_{CH_4} w$. There is a misprint in the definition of q in [9, p. 226]; the right-hand side of that definition should be multiplied by the heat release Q .

The computational domain extends radially from $r = 0.0$ cm to $r = 3.0$ cm and axially from $z = 0.0$ cm to $z = 25$ cm, as in [9, 42]. The initial (nonuniform) coarse grid is chosen to be more finely spaced in the region above the inner jet, because it is known that the flame forms in that area. The exact r - and z -coordinates of the initial grid are given in Table 3. We apply the same boundary conditions as used by Bennett and Smooke [9, 42].

4.2.1 Discretization of the partial differential equations

Due to the nature of the LDC method, PDEs (16)–(20) need only be discretized on tensor-product grids. We apply the following standard finite difference stencils at interior points (φ_{ij} denotes

the approximation of the value of φ at grid point (r_i, z_j) :

$$\left. \frac{\partial \varphi}{\partial r} \right|_{r_i, z_j} \doteq \frac{\varphi_{i+1,j} - \varphi_{i-1,j}}{r_{i+1} - r_{i-1}}, \quad (23)$$

$$\left. \frac{\partial^2 \varphi}{\partial r^2} \right|_{r_i, z_j} \doteq \frac{2\varphi_{i+1,j}}{(r_{i+1} - r_i)(r_{i+1} - r_{i-1})} - \frac{2\varphi_{ij}}{(r_{i+1} - r_i)(r_i - r_{i-1})} + \frac{2\varphi_{i-1,j}}{(r_i - r_{i-1})(r_{i+1} - r_{i-1})}, \quad (24)$$

$$\left. \frac{\partial^2 \varphi}{\partial r \partial z} \right|_{r_i, z_j} \doteq \frac{\varphi_{i+1,j+1} - \varphi_{i+1,j-1} - \varphi_{i-1,j+1} + \varphi_{i-1,j-1}}{(r_{i+1} - r_{i-1})(z_{j+1} - z_{j-1})}, \quad (25)$$

$$\left. \frac{\partial}{\partial r} \left(a \frac{\partial \varphi}{\partial r} \right) \right|_{r_i, z_j} \doteq \frac{2a_{i+1/2,j}(\varphi_{i+1,j} - \varphi_{ij})}{(r_{i+1} - r_i)(r_{i+1} - r_{i-1})} - \frac{2a_{i-1/2,j}(\varphi_{ij} - \varphi_{i-1,j})}{(r_i - r_{i-1})(r_{i+1} - r_{i-1})}, \quad (26)$$

$$\begin{aligned} \left. \frac{\partial}{\partial r} \left(a \frac{\partial \varphi}{\partial z} \right) \right|_{r_i, z_j} &\doteq \frac{a_{i+1/2,j}(\varphi_{i+1,j+1} + \varphi_{i,j+1} - \varphi_{i+1,j-1} - \varphi_{i,j-1})}{(r_{i+1} - r_{i-1})(z_{j+1} - z_{j-1})} \\ &\quad - \frac{a_{i-1/2,j}(\varphi_{i,j+1} + \varphi_{i-1,j+1} - \varphi_{i,j-1} - \varphi_{i-1,j-1})}{(r_{i+1} - r_{i-1})(z_{j+1} - z_{j-1})}. \end{aligned} \quad (27)$$

Axial derivatives are treated similarly. First-order upwinding is used on convective terms.

At the outer radial and at the outflow boundaries, the dependent variables approach constant values, so discretization errors are still small if one uses first-order-accurate discretizations to discretize the boundary conditions at these locations. However, a proper treatment of derivative boundary conditions is important at the other two boundaries to prevent bad overall accuracy due to large discretization errors at the boundary. Details can be found in [9, Sec. 4.2].

4.2.2 Solving the discrete system

The discretized governing equations and boundary conditions form a system of N_{eq} equations, where N_{eq} equals the number of grid points multiplied by the number of dependent variables. To linearize this system, we use a damped, modified Newton's method [19, 37] with a nested Bi-CGSTAB linear algebra solver due to Van der Vorst [43]; the latter is preconditioned using a block Gauss-Seidel preconditioner [20]. Note that it is far from trivial to find an initial guess for the Newton solution procedure on the initial grid. For the initial coarse tensor-product grid, we use a solution already calculated by Bennett and Smooke [9]. They construct a crude initial guess by using the prescribed boundary conditions at the inflow boundary and by duplicating these at all other horizontal levels. Via a pseudo time-stepping technique [39], they bring this initial guess into the convergence domain of Newton's method and then solve the steady problem.

4.2.3 Numerical results

Four different LDC simulations have been carried out on a 175 MHz SGI Octane with 1 GB of RAM. Each simulation starts from an already converged solution on the base tensor-product grid as described in Section 4.2.2. For each first solve on a local fine grid, we use as an initial guess the approximation found by interpolating the approximation on the parent grid, and pseudo time-stepping brings this guess into the Newton convergence domain. We take 50 adaptively chosen time steps; the initial time step is $\Delta t = 1 \cdot 10^{-6}$. At each level with more than one grid, 5 domain decomposition iterations (with no time-stepping) are done to improve the boundary conditions at the internal interfaces. The flagging of high activity boxes (see Section 3.1) is based on the spatial roughness of the methane mass fraction Y_{CH_4} , and $\epsilon = 2$ at all levels. On the initial grid,

$N_{\text{layer}} = 10$, and on finer grids, N_{layer} is chosen according to (6), (7). The refinement factor σ is 2, and the amount of overlap N_{dd} in the domain decomposition algorithm is set to 5 on all levels.

Plots of all dependent variables are shown in Figure 7. For comparison purposes, the left half of each of the nine plots shows the (old) LRR results of Bennett [42] and Bennett and Smooke [9]; the right half shows the (new) LDC results. The LDC results are calculated on a composite grid based on Table 3’s tensor-product grid with three additional refinement levels, consisting of two, four, and eight subgrids, respectively. In the high activity regions, the finest spacings in the LRR and LDC grids are identical. Beyond a slight difference in flame length (LRR: 0.715 cm; LDC: 0.720 cm), the data for the LRR and LDC methods display excellent agreement. (Note that flame length will be defined later in this section and will be discussed in more detail then.) Within both datasets, all dependent variables except for Y_{N_2} have large gradients in the reaction zone. Figure 8 shows the projection of Y_{CH_4} on LDC composite grids with increasingly fine resolution. Although the flame structure is similar in each plot, the flame length increases, with the largest increase occurring when the first refinement level is added. We will analyze this phenomenon at the end of this section, after we have compared the LRR and LDC algorithms.

In the LRR method, an unstructured grid is constructed from an initial tensor-product grid by flagging and refining high activity boxes individually. Unlike tensor-product grids, grid lines in an LRR grid are not required to extend from one domain boundary to the other. At points along interfaces between coarse and fine grids, special discretization stencils are used. A Newton solver is subsequently applied to the discretized PDE system on the complete unstructured grid.

An important difference between the LRR algorithm and the LDC algorithm with domain decomposition is that all equations at all grid points are solved simultaneously in the LRR approach, whereas the equations in the LDC approach are decoupled in two ways. First, coarse grid solves and fine grid solves are separated, and the approximations found in the different solves are coupled via the LDC iteration. Second, overlapping subgrids can be used at a given refinement level, and the approximations on the subgrids are coupled via the domain decomposition iteration.

Another difference between LRR and LDC is that because the LRR composite grid is unstructured, only high activity boxes and their neighboring boxes are refined. In the LDC approach, we have chosen to refine within rectangular grids only. Although this choice may cause the refinement of unflagged boxes, we can use simple data structures as well as standard discretization stencils at all grid points. LDC’s usage of rectangular grids explains why the *total* number of grid points in the LDC simulations is larger than in the LRR simulations, *cf.* Table 4. In the LDC simulations, however, the Newton solver is applied to many small grids individually rather than to one large grid. As shown in Table 5, the biggest tensor-product grid in the LDC simulation for $l_{\text{max}} = 3$ has only 16,653 points — a substantial memory savings over the single tensor-product grid of 312,872 points as well as over the LRR grid of 126,393 points.

Although the LDC method saves memory, its current implementation proves more expensive with respect to CPU time than the LRR algorithm. Without taking advantage of LDC’s inherent parallelization opportunities and without communicating more efficiently than by files, it is hard to make a fair comparison. Moreover, LDC needs a large number (5) of domain decomposition iterations per level due to the strong coupling of the Bunsen flame equations. Future research will be directed at using more advanced domain decomposition methods. Alternatively, at the price of a slightly more difficult data structure, one could develop a technique with nonrectangular fine grids and still employ LDC for the coupling between refinement levels.

Figure 7: Results for the Bunsen flame calculated with the LRR method (left half of each figure) and the LDC method with $l_{\max} = 3$ (right half). The finest spacings for the LRR and LDC grids are identical in the high activity regions. The area shown is the portion of the solution domain near the burner surface.

(a) Maximum level 0.

(b) Maximum level 1.

(c) Maximum level 2.

(d) Maximum level 3.

Figure 8: Plots of the methane mass fraction on the finest level for the LDC simulations with various values of $l_{\max}$.

LDC		LRR		ETP	
Grid points	L_f (cm)	Grid points	L_f (cm)	Grid points	L_f (cm)
–	–	4,060	0.480	4,060	0.480
9,408	0.580	6,235	0.580	9,408	0.580
22,344	0.640	13,280	0.640	26,730	0.640
62,804	0.680	37,642	0.690	88,683	0.685
189,760	0.720	126,393	0.715	312,872	–

Table 4: Numerical results for the Bunsen flame problem found with the local defect correction method (first column) and with the local rectangular refinement method (second column). The third column lists results found on equivalent tensor-product grids.

Level	Grid	Grid points	Sum	Total
0	bunsenfine	9,408	9,408	9,408
1	bunsenfine.1	7,085	14,246	23,654
	bunsenfine.2	7,161		
2	bunsenfine.1.1	10,075	40,776	64,430
	bunsenfine.1.2	10,125		
	bunsenfine.2.1	10,375		
	bunsenfine.2.2	10,201		
3	bunsenfine.1.1.1	16,303	125,330	189,760
	bunsenfine.1.1.2	16,065		
	bunsenfine.1.2.1	14,673		
	bunsenfine.1.2.2	14,553		
	bunsenfine.2.1.1	15,355		
	bunsenfine.2.1.2	15,189		
	bunsenfine.2.2.1	16,653		
	bunsenfine.2.2.2	16,539		

Table 5: Detailed information on the LDC composite grid for the Bunsen flame problem, for the simulation with $l_{\max} = 3$.

To verify the accuracy of the LRR results, including the predicted flame length, Bennett and Smooke [9] have created equivalent tensor-product (ETP) grids for each of the unstructured LRR grids by extending all grid lines to the domain boundaries. They define flame length L_f (see Figure 6) as the minimum z -coordinate on the centerline for which $Y_{\text{CH}_4} < 1 \cdot 10^{-4}$. In Table 4, we list the L_f values found in our LDC calculations (first column), and we quote LRR (second column) and ETP (third column) results from [9, Table 2]. The first LDC entry is blank; the LDC simulations have been started with a tensor-product grid whose resolution corresponds to that of the LRR grid with one level of refinement. This approach was used because the coarse (tensor-product) LRR initial grid could not adequately represent the LDC’s coarse grid solution after it had been updated by an LDC correction solve. The last ETP entry is also missing, because this ETP grid had too many grid points, exceeding the available computer memory [9].

Our LDC results are similar to both the LRR and ETP results. Bennett and Smooke [9] have also observed the same dependence of L_f on grid resolution in further LRR simulations in which the grid adaptation is based not only on Y_{CH_4} , but on Y_{CH_4} and T or on all dependent variables; see [9, Tables 3 and 4]. To explain this phenomenon, they [9, 42] begin with the result of Mikhel’son (as cited in [46]) that the laminar flame speed v_{lam} is proportional to the square root of the mixture-averaged diffusion coefficient D [9, form. (20)]. The first-order upwinding of the convective terms adds artificial diffusion, causing D to be overestimated. Refinement of the grid reduces the artificial diffusion, producing a lower effective D and a lower v_{lam} . In Figure 9(a) (this figure corresponds to [9, Figure 10]), we see that a lower v_{lam} causes the angle γ between the z -axis and the flame front to shrink. Since the radius r of the base of the flame is fixed, L_f must then decrease; refer to Figure 9(b).

We now carry out a short numerical analysis to support this qualitative explanation. To this end, we set $D(H) = D + D_{\text{art}}(H)$, with $D(H)$ the diffusion coefficient on a grid with typical spacing H ,

Figure 9: Idealized structure of the Bunsen flame.

D the real diffusion coefficient, and $D_{\text{art}}(H)$ the artificial diffusion introduced by the upwind scheme. The latter quantity is proportional to H , viz. $D_{\text{art}}(H) \doteq CH$, with C constant. This expression shows that $D_{\text{art}}(H)$ is smaller on finer grids, causing $D(H)$ to tend to D . Note that the amount of artificial diffusion could also be decreased by using a different discretization scheme for the convective terms, such as the modified Thiart scheme by Van 't Hof *et al.* [44].

As stated above, v_{lam} is proportional to $\sqrt{D(H)}$, so that (cf. [9, form. (20)])

$$v_{\text{lam}}(H) \approx \alpha \sqrt{D(H)}, \quad (28)$$

with α constant. For the angle γ between the z -axis and the flame front, we have

$$\sin \gamma(H) = \frac{v_{\text{lam}}(H)}{v_{\text{inflow}}}, \quad \tan \gamma(H) = \frac{r}{L_f(H)}. \quad (29)$$

In (29), v_{inflow} is the inflow velocity of the gas mixture, and r is the radius of the base of the flame. With some simple trigonometry, we find $\sin \gamma(H) = r / \sqrt{r^2 + L_f^2(H)}$. We set $\beta(H) := r^2 / r^2 + L_f^2(H)$. and find

$$\begin{aligned} \frac{\beta(H) - \beta(H/2)}{\beta(H/2) - \beta(H/4)} &= \frac{\sin^2 \gamma(H) - \sin^2 \gamma(H/2)}{\sin^2 \gamma(H/2) - \sin^2 \gamma(H/4)} \\ &= \frac{v_{\text{lam}}^2(H) - v_{\text{lam}}^2(H/2)}{v_{\text{lam}}^2(H/2) - v_{\text{lam}}^2(H/4)} \approx \frac{D_{\text{art}}(H) - D_{\text{art}}(H/2)}{D_{\text{art}}(H/2) - D_{\text{art}}(H/4)} \doteq 2. \end{aligned} \quad (30)$$

To verify the asymptotic behavior predicted by (30), we must evaluate the quotient on the left-hand side. Estimating r by the inner jet radius r_1 (see Figure 6, where $r_1 = 0.50$ cm) and using the L_f values found in the our LDC simulations as well as L_f from the coarsest tensor-product grid, cf. Table 4, we can calculate three values for the quotient. These values, listed in Table 6, agree reasonably well with the value of 2 as predicted by (30).

H	$L_f(H)$	$\beta(H)$	$\beta(H) - \beta(H/2)$	ratio
H_0	0.48	0.5204		
$H_0/2$	0.58	0.4263	0.0941	
$H_0/4$	0.64	0.3790	0.0473	1.9883
$H_0/8$	0.68	0.3509	0.0281	1.6843
$H_0/16$	0.72	0.3254	0.0256	1.0984

Table 6: Numerical verification of the relation between the grid spacing and the flame length.

From our analysis, we expect the flame length to increase the most when the first level of refinement is added. It should still increase, but less rapidly, with additional refinement. This trend holds for the first three values of ΔL_f , which are 0.1 cm, 0.06 cm, and 0.04 cm, respectively. The next value of ΔL_f is again 0.04 cm, however, which breaks the pattern. This fact is reflected in the final quotient (1.0984) in Table 6, which deviates further from 2 than the other two quotients listed. Although simulation results on finer grids are unavailable, (30) enables us to predict L_f on grids with even higher resolutions. Carrying out the extrapolation on the available data and assuming that (30) holds, we find that the Bunsen flame length would reach a limit of $L_f = 0.76$ cm.

5 Conclusions

In this paper, we have first extended the standard LDC method by including multilevel adaptive gridding, domain decomposition, and regridding, and we have then used this method to solve a system of PDEs on a locally refined composite grid. Actual computations take place on tensor-product grids only, each of which has grid spacings in agreement with the local roughness of the continuous solution. To choose the location of fine grids automatically, our method flags grid boxes for refinement based on a positive weight function w (a measure of solution roughness), and it also flags neighboring boxes to prevent the solution from becoming artificially trapped at grid interfaces. Because refinement of boxes by a factor depending on w would lead to unstructured grids, we use a constant refinement factor between successive grid levels, and we also employ more than one level. To form a subgrid that covers all flagged boxes at a given level, we formulate a heuristic approach that constructs a set of overlapping rectangles having the lowest cost, as calculated from a proposed cost function. Finally, the introduction of local refinement (and the subsequent solution procedure on the updated grid) may cause the area of high activity to move, so our method also incorporates a regridding strategy.

We illustrate the performance of the extended LDC algorithm with two applications. First, LDC is used to solve a heated plate problem whose exact solution is known. The domain decomposition heuristics produce a satisfactory set of rectangles, and the accuracy of the composite grid solution is comparable to that on a uniform tensor-product grid with spacing equal to that of the finest LDC grid. Execution times are heavily influenced by the file-intensive implementation, which was chosen for reasons of modularity. Memory usage for each LDC grid is extremely low compared to the equivalent tensor-product grid. Second, LDC is applied to a lean axisymmetric laminar Bunsen flame with one-step chemistry, which cannot be solved analytically. The simulations show the same characteristics as for the first problem: the method builds a good grid; the results show excellent agreement with those previously published in the literature; execution times are longer than comparable structured grid simulations; and memory usage is very low. Although the flame structure does not change as the grid undergoes refinement, the length of the Bunsen flame increases — a phenomenon that we quantify through a short numerical analysis.

In the future, we would like to investigate the possible inclusion of some more sophisticated domain decomposition techniques within the extended LDC method, so that fewer iterations will be required among grids at a given level, particularly for strongly coupled problems such as the Bunsen flame. Once those techniques are implemented, we plan to examine combustion problems with complex chemical mechanisms and multicomponent transport models. Finally, as there are virtually no conceptual hurdles in expanding the approach to higher dimensions, our ultimate goal is to apply the extended LDC method to three-dimensional combustion problems, for which its low memory usage and parallelization opportunities will play an important role.

Acknowledgments

The second and third authors gratefully acknowledge support from the US Department of Energy under Grant Number DE-FG02-88ER-13966.

References

- [1] M. J. H. Anthonissen. *Local defect correction techniques: analysis and application to combustion*. PhD thesis, Eindhoven University of Technology, Eindhoven, 2001.
- [2] M. J. H. Anthonissen, R. M. M. Mattheij, and J. H. M. ten Thije Boonkamp. Convergence analysis of the local defect correction method for diffusion equations. *Numerische Mathematik*, 95(3):401–425, 2003.
- [3] M. J. H. Anthonissen, B. van 't Hof, and A. A. Reusken. A finite volume scheme for solving elliptic boundary value problems on composite grids. *Computing*, 61:285–305, 1998.
- [4] D. C. Arney and J. E. Flaherty. An adaptive local mesh refinement method for time dependent partial differential equations. *Appl. Numer. Math.*, 5:257–274, 1989.
- [5] R. Becker, M. Braack, and R. Rannacher. Numerical simulation of laminar flames at low mach number by adaptive finite elements. *Combust. Theory Modelling*, 3:503–534, 1999.
- [6] J. Bell, M. Berger, J. Saltzman, and M. Welcome. Three-dimensional adaptive mesh refinement for hyperbolic conservation laws. *SIAM J. Sci. Comput.*, 15(1):127–138, Jan. 1994.
- [7] B. A. V. Bennett, J. Fielding, R. J. Mauro, M. B. Long, and M. D. Smooke. A comparison of the structures of lean and rich axisymmetric laminar Bunsen flames: application of local rectangular refinement solution-adaptive gridding. *Combust. Theory Modelling*, 3:657–687, 1999.
- [8] B. A. V. Bennett, C. S. McEnally, L. D. Pfefferle, and M. D. Smooke. Computational and experimental study of axisymmetric coflow partially premixed methane/air flames. *Combustion and Flame*, 123:522–546, 2000.
- [9] B. A. V. Bennett and M. D. Smooke. Local rectangular refinement with application to axisymmetric laminar flames. *Combust. Theory Modelling*, 2:221–258, 1998.
- [10] B. A. V. Bennett and M. D. Smooke. Local rectangular refinement with application to nonreacting and reacting fluid flow problems. *J. Comput. Phys.*, 151:684–727, 1999.
- [11] B. A. V. Bennett and M. D. Smooke. The effect of overall discretization scheme on Jacobian structure, convergence rate, and solution accuracy within the local rectangular refinement method. *Numerical Linear Algebra with Applications*, 8:513–536, 2001.
- [12] M. J. Berger and P. Colella. Local adaptive mesh refinement for shock hydrodynamics. *J. Comput. Phys.*, 82:64–84, 1989.
- [13] M. J. Berger and J. Olinger. Adaptive mesh refinement for hyperbolic partial differential equations. *J. Comput. Phys.*, 53:484–512, 1984.
- [14] B. J. Boersma, M. N. Kooper, F. T. M. Nieuwstadt, and P. Wesseling. Local grid refinement in large-eddy simulations. *J. Eng. Math.*, 32:161–175, 1997.
- [15] P. J. Coelho and J. C. F. Pereira. Calculation of a confined axisymmetric laminar diffusion flame using a local grid refinement technique. *Combust. Sci. and Tech.*, 92:243–264, 1993.
- [16] T. P. Coffee, A. J. Kotlar, and M. S. Miller. The overall reaction concept in premixed, laminar, steady-state flames, I. Stoichiometries. *Combust. Flame*, 54:155–224, 1983.
- [17] M. S. Day and J. B. Bell. Numerical simulation of laminar reacting flows with complex chemistry. *Combust. Theory Modelling*, 4:535–556, 2000.
- [18] H. C. de Lange and L. P. H. de Goeij. Numerical flow modelling in a locally refined grid. *International Journal for Numerical Methods in Engineering*, 37:497–515, 1994.
- [19] P. Deuflhard. A modified Newton method for the solution of ill-conditioned systems of nonlinear equations with application to multiple shooting. *Num. Math.*, 22:289–315, 1974.
- [20] A. Ern, V. Giovangigli, D. E. Keyes, and M. D. Smooke. Towards polyalgorithmic linear system solvers for nonlinear elliptic problems. *SIAM J. Sci. Comput.*, 15:681–703, 1994.
- [21] R. E. Ewing. Adaptive grid refinements for transient flow problems. In J. E. Flaherty, P. J. Paslow, M. S. Shephard, and J. D. Vasilakis, editors, *Adaptive Methods for Partial Differential Equations*, pages 194–205, Philadelphia, 1989. SIAM.
- [22] R. E. Ewing, R. D. Lazarov, and P. S. Vassilevski. Local refinement techniques for elliptic problems on cell-centered grids. I: Error analysis. *Math. Comp.*, 56:437–461, 1991.
- [23] P. J. J. Ferket. *Solving boundary value problems on composite grids with an application to combustion*. PhD thesis, Eindhoven University of Technology, Eindhoven, 1996.

- [24] P. J. J. Ferket and A. A. Reusken. Further analysis of the local defect correction method. *Computing*, 56:117–139, 1996.
- [25] P. J. J. Ferket and A. A. Reusken. A finite difference discretization method on composite grids. *Computing*, 56:343–369, 1996.
- [26] J. E. Flaherty, P. K. Moore, and C. Ozturan. Adaptive overlapping grid methods for parabolic systems. In J. E. Flaherty, P. J. Paslow, M. S. Shephard, and J. D. Vasilakis, editors, *Adaptive Methods for Partial Differential Equations*, pages 176–193, Philadelphia, 1989. SIAM.
- [27] W. D. Gropp and D. E. Keyes. Domain decomposition with local mesh refinement. *SIAM J. Sci. Stat. Comput.*, 13:967–993, 1992.
- [28] W. Hackbusch. Local defect correction and domain decomposition techniques. In K. Böhmer and H. J. Stetter, editors, *Defect Correction Methods. Theory and Applications, Computing, Suppl. 5*, pages 89–113, Wien, New York, 1984. Springer.
- [29] S. McCormick. Fast adaptive composite grid (FAC) methods: Theory for the variational case. In K. Böhmer and H. J. Stetter, editors, *Defect Correction Methods. Theory and Applications, Computing, Suppl. 5*, pages 115–121, Wien, New York, 1984. Springer.
- [30] S. McCormick and U. Rude. A finite volume convergence theory for the fast adaptive composite grid methods. *Appl. Numer. Math.*, 14:91–103, 1994.
- [31] S. F. McCormick. *Multilevel Adaptive Methods for Partial Differential Equations*, volume 6 of *Frontiers in Applied Mathematics*. SIAM, Philadelphia, 1989.
- [32] S. F. McCormick and J. Thomas. The fast adaptive composite grid (FAC) method for elliptic equations. *Math. Comp.*, 46:439–456, 1986.
- [33] S. Nefedov. Simulation of glass flow in an oven. In L. Arkeryd, J. Bergh, P. Brenner, and R. Pettersson, editors, *Progress in Industrial Mathematics at ECMI 98*, pages 106–113, Leipzig, 1999. B. G. Teubner Stuttgart.
- [34] V. Nefedov and R. M. M. Mattheij. Local defect correction with different grid types. *Numerical Methods for Partial Differential Equations*, 18:454–468, 2002.
- [35] R. B. Pember, L. H. Howell, J. B. Bell, P. Colella, W. Y. Crutchfield, W. A. Fiveland, and J. P. Jessee. An adaptive projection method for unsteady, low-mach number combustion. *Combust. Sci. Technol.*, 140:123–168, 1998.
- [36] Y. Saad. *Iterative Methods for Sparse Linear Systems*. PWS Publishing Company, Boston, 1996.
- [37] M. D. Smooke. Error estimate for the modified Newton method with applications to the solution of nonlinear, two-point boundary value problems. *J. Optim. Theory Appl.*, 39:489–511, 1983.
- [38] M. D. Smooke and B. A. V. Bennett. Numerical modeling of multidimensional laminar flames. In C. E. Baukal, Jr., V. Y. Gershtein, and X. Li, editors, *Computational Fluid Dynamics in Industrial Combustion*, pages 209–245, London, 2001. CRC Press.
- [39] M. D. Smooke, R. E. Mitchell, and R. J. Kee. Solution of premixed and counterflow diffusion flame problems by adaptive boundary value methods. In U. M. Ascher and R. D. Russell, editors, *Numerical Boundary Value ODEs*, pages 303–317, Boston, 1985. Birkhäuser.
- [40] L. M. T. Somers and L. P. H. de Goey. A numerical study of a premixed flame on a slit burner. *Combustion Science and Technology*, 108:121–32, 1995.
- [41] J. Thomas, R. Schweitzer, M. Heroux, S. McCormick, and A. M. Thomas. Application of the fast adaptive composite grid method to computational fluid dynamics. In *Numerical Methods in Laminar and Turbulent Flow*, pages 1071–1082, Swansea, 1987. Pineridge Press.
- [42] B. A. Valdati (Bennett). *Solution-adaptive gridding methods with application to combustion problems*. PhD thesis, Yale University, New Haven, CT, 1997.
- [43] H. A. van der Vorst. Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 13:631–644, 1992.
- [44] B. van ’t Hof, J. H. M. ten Thije Boonkkamp, and R. M. M. Mattheij. Discretization of the stationary convection-diffusion-reaction equation. *Numer Methods Partial Differential Eq*, 14:607–625, 1998.
- [45] J. U. Wappler. *Die lokale Defektkorrekturmethode zur adaptiven Diskretisierung elliptischer Differentialgleichungen mit finiten Elementen*. PhD thesis, Christian-Albrechts-Universität, Kiel, 1999. In German.
- [46] F. A. Williams. *Combustion Theory*. Addison-Wesley, New York, 1988.