Local Defect Correction Techniques: Analysis and Application to Combustion
Local Defect Correction Techniques: Analysis and Application to Combustion

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de Technische Universiteit Eindhoven, op gezag van de Rector Magnificus, prof.dr. M. Rem, voor een commissie aangewezen door het College voor Promoties in het openbaar te verdedigen op woensdag 25 april 2001 om 16.00 uur

door

Martijn Johannes Hermanus Anthonissen

gleboren te Breda
Dit proefschrift is goedgekeurd door de promotoren:

prof.dr. R.M.M. Mattheij
en
prof.dr. A.A. Reusken

Copromotor: dr.ir. J.H.M. ten Thije Boonkkamp
Preface

Perhaps I could best describe my experience of doing mathematics in terms of entering a dark mansion. One goes into the first room and it's dark, completely dark, one stumbles around bumping into the furniture and then gradually you learn where each piece of furniture is, and finally after six months or so you find the light switch, you turn it on, suddenly it's all illuminated, you can see exactly where you were.

Andrew Wiles
Fermat's Last Theorem, Horizon (BBC)

During the last couple of years of my mathematics study at Eindhoven University of Technology, I had the pleasure to be involved in two research projects. One of these projects was my graduation work that I carried out under the supervision of prof.dr.ir. J. de Graaf. The other research involved the symbolic calculation of the scattering coefficient in diffraction by a circular disk by use of Mathematica; this was joint work with prof.dr. J. Boersma. In both these projects, dr.ir. J.K.M. Jansen gave many helpful comments and advises.

Directly after my graduation, I participated in the Japan Prizewinners Program, a one-year postgraduate course financed by the Dutch government. Together with nineteen people like me, just graduated and fresh from university, I took part in the very first year of the program. It was a great opportunity and a fantastic experience to learn about Japan, study Japanese, live in Tokyo, and meet many new friends, both Dutch and Japanese.

Upon returning in Holland, I visited my old university after having been away from both Eindhoven and mathematics for more than a year. Realizing how much joy the previous scientific research had given me, the decision to start as a PhD student came naturally. Even though the university and faculty remained the same, the research group I joined was new.

My PhD research in the Scientific Computing Group has resulted in this thesis. Many people have been important in the realization of this book. In the first place, I would like to express my gratitude to prof.dr. R.M.M. Mattheij and to dr.ir. J.H.M. ten Thije Boonkkamp. Our frequent discussions have been indispensable for many results in this
thesis. Especially their support when we were trying to get a better understanding of the convergence behavior of the LDC method (which has resulted in Chapter 5) is greatly appreciated.

I got many helpful comments and interesting suggestions from prof.dr. A.A. Reusken. In particular, I would like to thank him for his remarks on an early version of Chapter 5 as well as the joint work on the generalization of the LDC algorithm to finite volume discretizations that has resulted in Chapter 4. For the latter work, I would also like to thank dr.ir. B. van ‘t Hof. My discussions with prof.dr. L.P.H. de Goey about a draft version of my thesis have resulted in many improvements, for which I am thankful.

Part of my research was carried out at Yale University. I am grateful to prof. Mattheij for introducing me to the research group of prof.dr. M.D. Smooke, and I would like to thank prof. Smooke as well as the people working in his group for their hospitality and support. I sincerely appreciate the help from dr. B.A.V. Bennett and her useful comments on my thesis. I thank Mike Noskov for our discussions and for showing me around New Haven.

I have enjoyed working in the Scientific Computing Group over the years, and I would like to thank my colleagues for the pleasant working environment. Among them, I would like to name my fellow PhD students Wienand Drenth, Arjan Frijns, Konstantin Laevsky, Bas van der Linden, and Kaichun Wang.

Finally, I would like to express my great appreciation to my family and friends for their support over the years. It would not have been possible to complete this thesis without the love and trust of my parents, Rien & Jenny, and my sister and her family, Esther, Ronald, Roy & Nikki. Work and study tend to consume a great part of my time—I would like to thank Ramon Clout, Arjan Frijns, and Roy Willemen for the weekly dinner parties, our theater visits as well as our routine of swimming three mornings each week. These activities have been and continue to be a good way for relaxation.
Contents

1 Introduction .................................................. 1
  1.1 Background ............................................. 1
  1.2 Outline of this thesis ..................................... 4

2 Mathematical model for a stationary laminar flame .............. 11
  2.1 Formulation of the conservation laws .................... 11
  2.2 Governing equations in cylindrical coordinates ............ 15
  2.3 Application problem: Bunsen flame ....................... 17

3 A two-grid local defect correction method ...................... 21
  3.1 An introductory example .................................. 21
  3.2 Formulation of the LDC method ........................... 24
  3.3 Some properties .......................................... 29

4 An LDC method for finite volume discretizations ............... 33
  4.1 Problem formulation and finite volume discretization on a uniform grid 34
  4.2 An iterative finite volume discretization on composite grids ........ 36
    4.2.1 Composite grid .................................... 36
    4.2.2 A local defect correction finite volume discretization ....... 37
    4.2.3 Properties of the LDC method .................... 42
  4.3 Numerical experiments .................................... 44
    4.3.1 A two-dimensional interface problem ................. 45
    4.3.2 A one-dimensional convection-diffusion problem ....... 47

5 Convergence behavior of the LDC method as an iterative process .... 49
  5.1 Expression for the iteration matrix ........................ 50
  5.2 Norm of the iteration matrix—continuous local problem ....... 52
    5.2.1 Trigonometric interpolation ........................ 53
    5.2.2 Norm of the iteration matrix ....................... 55
    5.2.3 Treatment of a non-zero value at the corner of the interface .. 59
  5.3 Norm of the iteration matrix—discrete local problem ......... 61
    5.3.1 Norm of the iteration matrix ....................... 62
    5.3.2 Treatment of a non-zero value at the corner of the interface .. 68
  5.4 Numerical experiments .................................... 70
Contents

6 Adaptive multi-level LDC in combination with domain decomposition 75
   6.1 Adaptive multi-level refinement ........................................ 76
   6.2 An example problem ...................................................... 82
   6.3 Domain decomposition .................................................... 84
   6.4 An example problem ...................................................... 88
   6.5 Regridding ................................................................. 92

7 Numerical simulation of the Bunsen flame 95
   7.1 Discretization of the partial differential equations ................. 95
   7.2 Solving the discrete system ............................................ 97
   7.3 Numerical results ......................................................... 99

8 Conclusions and recommendations 107

Bibliography 112

Index 113

Summary 115

Samenvatting 117

Curriculum vitae 119
Chapter 1

Introduction

1.1 Background

Combustion processes occur in a large variety and are of fundamental importance both for industry and for ordinary daily life. For combustion applications, one may e.g. think of industrial burners, combustion engines as well as more simple household appliances such as (water) heating devices. Combustion processes are very complicated phenomena for a variety of reasons. It is, however, important to understand them well, as better knowledge may help to design more efficient burner systems. Another design goal is to develop burners that have a lower exhaust of polluting gases, thus adhering to increasingly strict governmental regulations. Computer simulations may be of help in the design of better burners for industrial and domestic use.

Because of the complexity of combustion processes, it is hard, if not impossible, even with present day computer technology to carry out detailed simulations in complex geometries. For this reason, many researchers study relatively simple flames, either by reducing the complexity of the chemical reactions taking place in the flame or by restricting their simulations to simple geometries. Typical examples of such geometries are one-dimensional flat flame burners, see e.g. [Somers, 1994], two-dimensional slit burners, see e.g. [Mallens, 1996, Somers, 1994, van ’t Hof, 1998], or axisymmetric Bunsen burners, see e.g. [Bennett and Smooke, 1998, Bennett and Smooke, 1999, Valdati, 1997]. The latter type is shown in Figure 1.1 and will also be studied in this thesis.

In the mathematical description of a flame, we consider it a flowing gas mixture in which chemical reactions take place. As such, a flame is described by physical quantities describing the flow, such as the density, velocity, and pressure, and by quantities describing the chemical reactions in the mixture, such as the mass fractions of the chemical species in the mixture, the enthalpy or energy, and the temperature. Conservation of the various physical quantities is governed by flow equations and combustion equations. The
flow equations express conservation of mass and momentum. The combustion equations express conservation of the species mass fractions and the enthalpy or energy.

The system of equations describing a laminar flame has several properties that make flame simulations a challenge even with state-of-the-art computers. All conservation laws are nonlinear partial differential equations. They may describe very complex flows, and the chemical source terms are strongly nonlinear. Detailed chemical models involve both many species as well as many chemical reactions. The chemical reactions may have very different time scales. This leads to numerical simulations with so many unknowns that they may well become prohibitively expensive.

Apart from large differences in time scales, there are also large differences in geometric scales. The preheat zone in a flame and the chemically active layer are much smaller than the computational domain. As an example, consider Figure 1.2. It shows only a small part of the computational domain used in a simulation carried out in this thesis for the Bunsen burner of Figure 1.1. Plotted are the temperature and the mass fraction of methane, the fuel in the reaction. The complete computational domain ranges from \( r = 0 \) cm to \( r = 3 \) cm in radial direction and from \( z = 0 \) cm to \( z = 25 \) cm in axial direction. The figure shows the area from \( r = 0 \) cm to \( r = 1 \) cm and from \( z = 0 \) cm to \( z = 2 \) cm. The methane mass fraction is almost constant below and above the flame zone, and decreases very rapidly across the flame zone, where the fuel is consumed by chemical reactions. It is evident, that a computational grid for a problem of this type should reflect the solution behavior, i.e., it should have many grid points with fine spacing in and near the flame zone, and it may be (much) coarser outside the chemically active
1.1 Background

(a) Temperature.

(b) Methane mass fraction.

Figure 1.2: A plot of the temperature and of the methane (fuel) mass fraction in a small part of the computational domain.

region. In this thesis, we will study a method to do numerical simulations on grids satisfying these restrictions.

Laminar flame simulation has been subject of research at Eindhoven University of Technology for a number of years. De Lange and De Goey [de Lange, 1992, de Lange and de Goey, 1994] present a method to discretize and solve the two-dimensional low Mach number flow equations on an adaptive locally refined non-staggered grid. Their discretization is based on a streamfunction-vorticity formulation. They use hybrid and power-law schemes rather than upwind schemes for the convective terms. In their refinement strategy, they add grid points to a coarse grid in areas where a given property has large gradients. Grid points are added half-way between coarse grid points. The algorithm is applied to a laminar premixed methane-air flame with simple chemistry.

Somers [Somers, 1994] has studied the effect of transport and chemical models on lean methane-air flames by numerical simulations of a premixed laminar flat flame with a detailed chemical model. He uses the same adaptive gridding strategy as De Lange. The simulated burner is a perforated metal plate with a cooling rim at the outlet of the burner. Near the center of the burner, the flame can be considered one-dimensional. Two-dimensional simulations are also carried out to study the deviations from one-dimensionality. Usage of detailed chemical models leads to both a large number of unknowns in simulations as well as to a stiff system of differential equations. To reduce the size of the problem as well as remove stiffness from the system of equations, reduction techniques can be applied. This means that one omits less important species or reactions or that one assumes a steady state for intermediate species or a partial equilibrium for certain reactions. Somers studies systematic reduction of detailed chemical
mechanisms. Eggels [Eggels, 1995] distinguishes fast and slow reaction groups, and makes partial equilibrium assumptions for the fastest reaction groups. He uses reduced chemical models to predict NO formation in flames.

Mallens [Mallens, 1996] explains how changes in burner design for small domestic appliances that are meant to reduce the exhaust of pollutant gases, may endanger flame stability. In his thesis, he studies flame instabilities, such as quenching, flash-back, and blow-off. He performs simulations of two-dimensional steady state laminar premixed methane-air flames on both slit and tube burners.

All of the previous studies use a finite volume discretization method due to Thiart [Thiart, 1990a, Thiart, 1990b]. Van ’t Hof [van ’t Hof, 1998, van ’t Hof et al., 1998] presents a modified Thiart scheme. This modified scheme is second order accurate for both diffusion and convection dominated flow, has no oscillations in regions with high convection and has only a three-point coupling. The author also shows the shortcomings of traditional pressure correction schemes for low Mach number laminar flames, and introduces a generalization in which the correction step is based on a so-called expansion equation in stead of the continuity equation traditionally used. Van ’t Hof carries out simulations of a time-dependent two-dimensional single slit burner. He uses both one-step and skeletal chemical models.

### 1.2 Outline of this thesis

We formulate the mathematical model for a stationary laminar flame in Chapter 2. As stated in the previous section, this model follows from the flow equations that express conservation of mass and momentum and from the combustion equations that express conservation of the species mass fractions and the energy. We use a vorticity-velocity formulation of the equations, and write the conservation laws in cylindrical coordinates. At the end of the chapter, we present a combustion problem: the two-dimensional axisymmetric Bunsen flame depicted in Figure 1.1. We will use a one-step chemistry model. This problem was previously presented by Bennett and Smooke in [Bennett and Smooke, 1998, Valdati, 1997]. Results of numerical simulations of the Bunsen flame will be presented in Chapter 7. Our simulations show that all dependent variables in the flame have large gradients in the flame zone. The flame zone forms only a small part of the computational domain.

Solutions that have highly localized properties do not only occur in combustion processes. Examples of partial differential equations with solutions that are rapidly varying functions of the spatial or temporal coordinates appear e.g. in shock hydrodynamics or transport in porous media too. For boundary value problems with solutions that have one or a few small regions with high activity, a fine grid is needed in regions with high activity, whereas a coarser grid would suffice in the rest of the domain. For this reason, the usage of a global uniform fine grid is computationally inefficient. An obvious choice would be to use a truly nonuniform refined grid.
However, uniform grids have several advantages over truly nonuniform grids: uniform grids can be represented by simple data structures, 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 for elliptic partial differential equations [Ewing, 1989], [Gropp and Keyes, 1992], [McCormick, 1989], hyperbolic partial differential equations [Arney and Flaherty, 1989], [Berger and Colella, 1989], and parabolic partial differential equations [Flaherty et al., 1989], [Trompert, 1994].

In Chapter 3 of this thesis, we consider a discretization method for elliptic boundary value problems introduced by Hackbusch [Hackbusch, 1984]. 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 grid sizes that cover different parts of the domain. At least one grid, the coarse grid, should cover the entire domain, and its grid size should be chosen in agreement with the relatively smooth behavior of the solution outside the high activity areas. Apart from this global coarse grid, one or several local fine grids are used which are also uniform. Each of the local grids covers only a (small) part of the domain and contains a high activity region. The grid sizes of the local grids are chosen in agreement with the behavior of the continuous solution in that part of the domain.

The LDC method 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 right hand side of the coarse grid problem. At each iteration step, the process yields a discrete approximation of 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.

An analysis of the LDC technique in combination with finite difference discretizations is presented in [Ferket, 1996], [Ferket and Reusken, 1996a], [Ferket and Reusken, 1996b]. The LDC method is combined with finite volume discretizations in [Anthonissen et al., 1998], [Anthonissen et al., 1999] and [van 't Hof, 1998] Ch. 7. We will also study the generalization of the LDC method to finite volume discretizations in Chapter 4. An application of the LDC algorithm in a finite volume context was presented by Nefedov [Nefedov, 1999]. He studies the feasibility of applying the LDC method for local grid refinement in numerical simulations of the flow and heat transfer in a glass tank. Finally, LDC is studied in combination with finite element discretizations in [Wappler, 1999].

We formulate the LDC algorithm for two-dimensional problems in Chapter 3. In a general setting, we prove some results for the fixed point of the iteration, assuming such a fixed point exists. For the model problem of Poisson’s equation on the unit square with zero Dirichlet boundary conditions, Ferket [Ferket, 1996] managed to derive an upper bound for the global discretization error of the composite grid implicitly given by the
Introduction

Figure 1.3: The discretization on the composite grid given by the LDC algorithm with the standard choice for the correction term (left figure) and with the finite volume adapted correction term (right figure). The two composite grid discretizations differ at the control volumes located at the interface between the coarse and fine grids.

LDC iteration. We quote his results at the end of the introductory LDC chapter.

In Chapters 4–6, we present generalizations of Hackbusch’s original LDC method and we analyze the algorithm. More specifically, Chapter 4 deals with the generalization of the LDC method to finite volume discretizations. In a straightforward generalization of the LDC algorithm as presented in Chapter 3, the discrete conservation property, which is one of the main attractive features of a finite volume method, does not hold for the composite grid approximation. We present a modified LDC method, which is based on a special form of the defect correction term used in the right hand side of the coarse grid problem.

Due to this finite volume adapted defect correction term, the conservation property is preserved in the discretization on the composite grid. The difference between the two composite grid discretizations is illustrated by Figure 1.3. The discretization on the left does not satisfy the conservation property in the control volumes that belong to coarse grid points located at the interface between the coarse and the fine grids. The composite grid discretization on the right of Figure 1.3 does preserve the conservation property at all control volumes.

Both in Chapter 3 and in Chapter 4, several properties are proved for the fixed point of the LDC iteration. The conservation property on the composite grid in the finite volume adapted LDC method, cf. Figure 1.3, holds for example for the limit LDC discretization. For this reason, it is important to study the convergence behavior of the LDC algorithm. This analysis is the topic of Chapter 5. We derive an expression for the iteration matrix of the algorithm. This expression is quite general. Next, we focus on the model problem of
Poisson's equation on the unit square with Dirichlet boundary conditions. It is assumed that the area of refinement is located at the lower left corner. If we use a special kind of interpolation on the interface between the fine and coarse grids, we can derive an upper bound for the (infinity) norm of the iteration matrix $M$ of the form

$$\|M\|_{\infty} \leq CH^2,$$  \hspace{1cm} (1.1)

in which $C$ is a constant and $H$ is the grid size of the global coarse grid. This is verified in some numerical experiments.

Discretization methods on composite grids have been discussed by other authors. McCormick presents the finite volume element (FVE) method, which is used in the fast adaptive composite grid (FAC) method (cf. [McCormick, 1989], [McCormick and Thomas, 1986], [Thomas et al., 1987]). Ewing, Lazarov and Vassilevski [Ewing, 1989], [Ewing et al., 1991] give an analysis of a finite volume based local refinement technique with composite grids. In both approaches, an explicit discretization scheme for the composite grid is proposed, in which special difference stars near the composite grid interfaces are used. The resulting discrete system is then solved by an iterative method (e.g. FAC) which may take advantage of the composite grid structure. This is a crucial difference with the LDC method, which combines standard discretizations on uniform grids only and does not use an a priori given composite grid discretization. For the FAC method in a variational setting, convergence results have been given by McCormick [McCormick, 1984]. The variational theory is extended to the finite volume element method in [McCormick and Rüde, 1994] by interpreting FVE as an approximate finite element scheme.

Boersma, Kooper, Nieuwstadt and Wesseling [Boersma et al., 1997] present a method for nested-grid calculations in a large eddy-simulation context. As in the LDC algorithm, their method uses a global coarse grid with one or more local fine grids to cover critical areas. The method uses second order accurate finite volume discretizations on the global and local grids, which are all uniform. Staggered grids are being used. The communication from the global grid to the local fine grids is via artificial Dirichlet boundary conditions. A special interpolation is used for the velocity components normal to the interface to ensure mass conservation. For this interpolation, the authors follow Thompson and Ferziger [Thompson and Ferziger, 1989]; they apply multi-linear interpolation followed by a correction to ensure that the sum of the mass flux through the fine grid cell faces that form a coarse grid cell face equals the original mass flux through the coarse grid cell face.

Chapter 6 deals with some extensions to the standard LDC method as studied in previous chapters. We present a technique to discretize and solve elliptic boundary value problems on composite grids found by adaptive grid refinement. In this technique, we will only use standard discretizations on rectangular tensor-product grids. The full algorithm is obtained by successively adding adaptivity, multi-level refinement, domain decomposition and regridding to the standard LDC algorithm as presented in Chapter 3. A procedure for adaptive gridding introduced by Bennett and Smooke [Bennett and Smooke, 1998], [Bennett and Smooke, 1999], [Valdati, 1997] is formulated and combined with the LDC algorithm. Based on a weight function, that measures the smoothness of the solution of the partial differential equation under consideration, high activity areas
are determined and flagged for refinement, cf. Figure 1.4.

The combination with LDC is new and is as follows. The flagged boxes are covered with a rectangular patch. In the patch, a finer grid is chosen. The boundary value problem is discretized and solved on the fine grid in the patch. When the problem has been solved on the finer grid, the more accurate solution found locally is used to estimate the local discretization error of the coarse grid discretization. Next, the approximation on the initial coarse tensor-product grid is updated using this estimate for the discretization error. The solution procedure may be applied recursively, i.e., the rectangular patches used to cover high activity areas in the coarse grid may be refined themselves. The maximum level of refinement can be chosen such that the desired accuracy is achieved on the composite grid.

Usage of a single rectangular patch to cover all flagged boxes in a grid may be inefficient. The reason is that it may cause refinement of a large number of unflagged boxes. To remedy this inefficiency as well as to prevent the grids from becoming too large, we combine the adaptive multi-level LDC algorithm with domain decomposition. Rather than refining the smallest enclosing rectangle of all flagged boxes in a grid, we use a set of overlapping rectangles to cover all boxes that have been flagged for refinement. The overlap of the rectangles is necessary in situations where a high activity region is cut into two or more rectangles. Interfaces between rectangles may intersect high activity zones. Large errors at these interfaces can be remedied by performing a number of domain decomposition iterations. Finally, we note that refining a grid and solving the boundary value problem on the new composite grid may cause the area of high activity to move. When this happens, areas of the grid may be refined unnecessarily or areas may not be refined whereas they do require refinement. Therefore, we formulate a regridding procedure.

In Chapter 7 of this thesis, we apply our proposed adaptive multi-level LDC algorithm to the Bunsen flame problem described above, cf. Figure 1.1. We outline the discretization of the system of partial differential equations with the finite difference method and sketch the solution process used on the individual tensor-product grids. The large discrete system of nonlinear equations that is found on each tensor-product grid after discretization is solved using a damped Newton method. The linearized systems are solved using preconditioned Bi-CGSTAB. A pseudo time-stepping technique is applied to bring an initial guess within the convergence domain of the Newton method.

Our simulation results show that all dependent variables except for the nitrogen mass fraction have large gradients in the flame zone. A remarkable characteristic of the Bunsen flame problem is that the size of the flame increases on the finer grids. The increase is strongest when the first level of refinement is added. The flame length still increases, but less rapidly, with additional refinement. The structure of the flame is similar on all grids. We verify the increase of the flame length using Richardson extrapolation.

The Bunsen flame problem was previously treated by Bennett [Valdati, 1997] and Bennett and Smooke [Bennett and Smooke, 1998]. In both references, simulations have been performed with the local rectangular refinement (LRR) method. In the LRR method, an
unstructured grid is constructed from an initial tensor-product grid by flagging and refining high activity boxes. The restriction that all grid lines should extend from one domain boundary to the other, as is the case for tensor-product grids, is lifted. At interior boundary points, i.e., points at interfaces between coarse and fine grids, special discretization stencils are applied. Next, the problem is solved on the complete unstructured grid. The results found by Bennett and Smooke are similar to our LDC results. In particular, the increase of the size of the Bunsen flame was also observed in [Bennett and Smooke, 1998, Valdati, 1997].

Adaptive gridding methods have been used by many authors in flame simulations. Coelho and Pereira [Coelho and Pereira, 1993] study a method for the numerical simulation of a two-dimensional axisymmetric methane-air diffusion flame. Their approach is based on a finite volume discretization of the governing equations on a non-staggered grid. The convective terms are discretized using the hybrid scheme. The area of local refinement is a priori chosen: the user has to select the control volumes that should be refined as well as the maximum level of refinement. The refinement factor is 2. Special care is taken to ensure conservation across the interface. As mentioned before, De Lange and De Goey [de Lange, 1992, de Lange and de Goey, 1994], Somers [Somers, 1994], and Mallens [Mallens, 1996] use a refinement strategy based on adding grid points in areas where a given property has large gradients. Braack [Braack, 1998] presents an adaptive method for the simulation of two-dimensional low Mach number flames that uses the finite element discretization method rather than the (in numerical combustion more common) finite difference and finite volume methods. His adaptive gridding is based on an a posteriori error estimator. Bennett and Smooke have successfully applied the LRR method to a number of combustion problems with both simple and detailed chemical
Introduction

Chapter 2

Mathematical model for a stationary laminar flame

2.1 Formulation of the conservation laws

In this section we will formulate the equations describing a stationary laminar flame. We will start by presenting the governing conservation laws as well as the constitutive relations. We consider a reacting gas mixture of \( N \) different chemical species, and we will formulate the equations for conservation of mass, momentum, and energy. Models will be presented for the diffusion velocities of the species, for the reaction rate, for the stress tensor, and for the heat flux vector. To simplify the system of equations, we will assume that the chemical behavior of the mixture can be described by one chemical reaction. The system of equations presented describes a steady flame. More detailed information on the derivation of the conservation equations for a reacting gas mixture can be found in [ten Thije Boonkkamp, 1993, Williams, 1985].

The reaction rate \( w_i (i = 1, 2, \ldots, N) \) of species \( i \) is defined as the mass of species \( i \) that is created (or destroyed) by chemical reactions per unit volume and per unit time. The reaction rate has physical dimension \( \text{kg}/(\text{m}^3 \text{s}) \). As no mass can be created or destroyed by chemical reactions, we have

\[
\sum_{i=1}^{N} w_i = 0. \tag{2.1}
\]

We will denote the mass density of the gas mixture by \( \rho \) (dimension \( \text{kg}/\text{m}^3 \)) and the mass density of species \( i \) by \( \rho_i \) (dimension \( \text{kg}/\text{m}^3 \)). The mass fraction \( Y_i \) of species \( i \) is defined as

\[
Y_i := \frac{\rho_i}{\rho}. \tag{2.2}
\]
By definition,
\[ \sum_{i=1}^{N} Y_i = 1. \]  
(2.3)

The flow velocity of species \( i \) is denoted by \( v_i \). With these definitions, the continuity equation for species \( i \) for a stationary laminar flame can be formulated as
\[ \nabla \cdot (\rho Y_i v_i) = w_i. \]  
(2.4)

The mass-weighted average flow velocity \( \bar{v} \) of the gas mixture is defined by
\[ \bar{v} := \sum_{i=1}^{N} Y_i v_i. \]  
(2.5)

The diffusion velocity of species \( i \) is defined as
\[ V_i := v_i - \bar{v}. \]  
(2.6)

As a model for the diffusion velocities of trace species, we will use a generalized Fick’s law, see [Majda and Lamb, 1991], viz.
\[ Y_i V_i = -D_i \nabla Y_i, \]  
(2.7)

for \( i = 1, 2, \ldots, N-1 \), in which \( D_i \) is the diffusion coefficient of species \( i \) in the mixture. The diffusion velocity of the abundant species, numbered \( N \), follows from the constraint
\[ \sum_{i=1}^{N} Y_i V_i = 0. \]  
(2.8)

Substitution of (2.6) and (2.7) into (2.4) gives (i = 1, 2, \ldots, N - 1)
\[ \nabla \cdot (\rho Y_i v_i) = \nabla \cdot (\rho D_i \nabla Y_i) + w_i. \]  
(2.9)

Summation of (2.4) over all species leads to the continuity equation for the mixture
\[ \nabla \cdot (\rho \bar{v}) = 0. \]  
(2.10)

For the reaction rates \( w_i \), we formulate a model based on chemical kinetics. We will assume that only one chemical reaction takes place. Although this assumption is generally not true, the global behavior of the flame may often be adequately described by a single global reaction. We represent the irreversible chemical reaction by
\[ \sum_{i=1}^{N} \nu'_i M_i \rightarrow \sum_{i=1}^{N} \nu''_i M_i, \]  
(2.11)

in which \( M_i \) denotes chemical species \( i \) and \( \nu'_i \) and \( \nu''_i \) denote the stoichiometric coefficients. For reaction (2.11), the reaction rate of species \( i \) equals
\[ w_i = W_i (\nu''_i - \nu'_i) k \prod_{j=1}^{N} \left( \frac{\rho Y_j}{W_j} \right)^{\gamma_j}, \]  
(2.12)
2.1 Formulation of the conservation laws

in which \( W_i \) denotes the molar weight of species \( i \) (dimension \( \text{kg/mol} \)), and \( k \) denotes the specific rate constant of the reaction. For the choice of the exponents \( \gamma_i \), we will use a model by Coffee, Kotlar and Miller [Coffee et al., 1983], see Section 2.3. In this model, we take all \( \gamma_i \) integer-valued, as opposed to e.g. De Lange [de Lange, 1992], who uses non-integer exponents. We assume that \( k \) satisfies Arrhenius’ law, viz.

\[
k = A \exp \left( -\frac{E}{RT} \right).
\]  

(2.13)

Here, \( A \) is the frequency factor of the reaction (dimension \( \text{mol/m}^3 \cdot \text{s} \)), \( E \) is the activation energy of the reaction (dimension \( \text{J/mol} \)), \( R \) is the universal gas constant (equal to \( 8.3147 \text{J/(mol K)} \)), and \( T \) is the temperature of the gas mixture (dimension K). Some authors include additional powers of the temperature in the right hand side of (2.13), see [Williams, 1985]. In our model, the rate constant \( k \) has the same dimension as the frequency factor \( A \).

If we assume that the only external force acting on the mixture is the gravity force, the momentum equations can be formulated as

\[
\nabla \cdot (\rho vv^T) = \nabla \cdot \sigma + \rho g.
\]  

(2.14)

In (2.14), \( \sigma \) is the stress tensor (dimension \( \text{N/m}^2 \)) and \( g \) is the gravitational acceleration. The stress tensor \( \sigma \) can be written as

\[
\sigma = -pI + \tau,
\]  

(2.15)

in which \( p \) is the hydrostatic pressure and \( \tau \) is the viscous stress tensor (both have dimension \( \text{N/m}^2 \)). We assume that the mixture behaves like a Newtonian fluid, i.e.,

\[
\tau = \mu \left( \nabla v + (\nabla v)^T - \frac{2}{3} (\nabla \cdot v) I \right),
\]  

(2.16)

where \( \mu \) is the viscosity coefficient of the gas mixture (dimension \( \text{kg/(m s)} \)). Substitution of (2.15) into (2.14) gives

\[
\nabla \cdot (\rho vv^T) = -\nabla p + \nabla \cdot \tau + \rho g,
\]  

(2.17)

which expresses conservation of momentum.

The last conservation law is the energy equation. Formulated in terms of the specific internal energy \( e \) of the gas mixture (dimension \( \text{J/kg} \)), it reads

\[
\nabla \cdot (\rho ev) = -\nabla \cdot q + \nabla \cdot (\sigma v) - (\nabla \cdot v) \cdot v.
\]  

(2.18)

In (2.18), \( q \) is the heat flux vector for the gas mixture (dimension \( \text{J/m}^2 \cdot \text{s} \)). We adopt the following model for \( q \), which incorporates heat transport due to conduction and enthalpy transport by diffusion:

\[
q = -\lambda \nabla T + \rho \sum_{i=1}^{N} h_i y_i V_i.
\]  

(2.19)
In (2.19), we have introduced $\lambda$, the \textit{thermal conductivity} of the gas mixture (dimension $J/(m\cdot s\cdot K)$) and $h_i$, the \textit{specific enthalpy} of species $i$ (dimension $J/kg$). The specific enthalpy $h_i$ is defined as

$$h_i := h_i^0 + \int_{T_0}^T c_{pi}(\xi) \, d\xi.$$  \hfill (2.20)

In (2.20), $h_i^0$ is the \textit{specific heat of formation} for species $i$ at the reference temperature $T_0$ (dimension $J/kg$), and $c_{pi}$ is the \textit{specific heat at constant pressure} for species $i$ (dimension $J/(kg\cdot K)$). Substitution of the models (2.15) and (2.19) into (2.18) gives

$$\nabla \cdot (\rho e v) = \nabla \cdot (\lambda \nabla T) - \nabla \cdot \left( \rho \sum_{i=1}^{N} h_i Y_i V_i \right) - p \nabla \cdot v + \tau : \nabla v,$$  \hfill (2.21)

where we have used $\tau : \nabla v$, defined by

$$\tau : \nabla v := \nabla \cdot (\tau v) - (\nabla \cdot \tau) \cdot v.$$  \hfill (2.22)

The energy equation can also be expressed in terms of the \textit{specific enthalpy} rather than the specific internal energy. We define the specific enthalpy $h$ of the gas mixture (dimension $J/kg$) by

$$h := e + \frac{p}{\rho}.$$  \hfill (2.23)

Reformulating (2.21) in terms of $h$, we find

$$\nabla \cdot (\rho h v) = \nabla \cdot (\lambda \nabla T) - \nabla \cdot \left( \rho \sum_{i=1}^{N} h_i Y_i V_i \right) + v \cdot \nabla p + \tau : \nabla v,$$  \hfill (2.24)

which expresses conservation of enthalpy.

Summarizing, we have found the system of conservation equations (2.9), (2.10), (2.17), and (2.24). This system consists for two-dimensional problems of $N+3$ independent equations. The independent variables are $\rho$, $v$, $Y_i$ ($i = 1, 2, \ldots, N-1$), $p$, $h$, and $T$. Note that only the first $N-1$ mass fractions are unknown, as $Y_N$ follows from (2.3). The number of independent variables equals $N+5$ for two-dimensional problems, so that we need two additional equations. The first equation we add to the system is the \textit{equation of state}, viz.

$$p = \rho RT \sum_{i=1}^{N} \frac{Y_i}{W_i}.$$  \hfill (2.25)

The last equation we formulate is the \textit{thermodynamic identity}

$$h = \sum_{i=1}^{N} Y_i h_i.$$  \hfill (2.26)

We will use the thermodynamic identity (2.26) to reformulate the energy equation (2.24) in terms of the temperature. We assume that the last two terms in the right hand side of (2.24) (the pressure term and the viscous term) can be neglected, which is realistic
2.2 Governing equations in cylindrical coordinates

for low-speed flow problems. If we introduce the specific heat at constant pressure of the mixture (dimension J/(kg K)) by

\[ c_p = \sum_{i=1}^{N} Y_i c_{pi}, \]  

(2.27)

then we have with (2.26)

\[ \nabla h = c_p \nabla T + \sum_{i=1}^{N} h_i \nabla Y_i. \]  

(2.28)

Combining (2.24) and (2.28), and using Fick’s law (2.7), (2.8) and the continuity equations (2.9), (2.10), we find

\[ c_p \rho v \cdot \nabla T = \nabla \cdot (\lambda \nabla T) + \sum_{i=1}^{N-1} \rho D_i \nabla (h_i - h_N) \cdot \nabla Y_i - \sum_{i=1}^{N-1} (h_i - h_N) w_i. \]  

(2.29)

We will assume that all \( c_{pi} \) \( (i = 1, 2, \ldots, N) \) are equal. Under this assumption the second term in the right hand side of (2.29) vanishes. The energy equation formulated in terms of the temperature then reads

\[ c_p \rho v \cdot \nabla T = \nabla \cdot (\lambda \nabla T) - \sum_{i=1}^{N-1} h_i^* w_i, \]  

(2.30)

with \( h_i^* := h_i - h_N \).

We conclude this section by listing the system of partial differential equations

\[ \nabla \cdot (\rho Y_i v) = \nabla \cdot (\rho D_i \nabla Y_i) + w_i, \quad i = 1, 2, \ldots, N - 1, \]  

(2.31)

\[ \nabla \cdot (\rho v) = 0, \]  

(2.32)

\[ \nabla \cdot (\rho vv^T) = -\nabla p + \nabla \cdot \tau + \rho g, \]  

(2.33)

\[ c_p \rho v \cdot \nabla T = \nabla \cdot (\lambda \nabla T) - \sum_{i=1}^{N-1} h_i^* w_i. \]  

(2.34)

Together with the equation of state (2.25) and the thermodynamic identity (2.26), this system consists of \( N + 5 \) equations for two-dimensional problems.

2.2 Governing equations in cylindrical coordinates

Since the application problem to be discussed in Chapter 7 is axisymmetric, we will formulate the governing equations in cylindrical coordinates \( r, \theta, z \). The species continuity equations (2.31) in cylindrical coordinates read \( (i = 1, 2, \ldots, N - 1) \)

\[ \rho v_r \frac{\partial Y_i}{\partial r} + \rho v_z \frac{\partial Y_i}{\partial z} = \frac{1}{Le_i} \left\{ \frac{1}{r} \frac{\partial}{\partial r} \left( r \frac{\lambda}{c_p} \frac{\partial Y_i}{\partial r} \right) + \frac{\partial}{\partial z} \left( \frac{\lambda}{c_p} \frac{\partial Y_i}{\partial z} \right) \right\} + w_i. \]  

(2.35)
In (2.35), we have used the Lewis number of species $i$, which is the ratio of thermal conduction to species diffusion:

$$Le_i := \frac{\lambda}{\rho D_i c_p}. \quad (2.36)$$

The Lewis numbers of all species are assumed to be constant. The reaction rate $w_i$ is given by (2.12).

We will use a vorticity-velocity formulation of the equations. Vorticity is formally defined as the curl of the velocity field. Since we will treat an axisymmetric problem, the vorticity vector $\mathbf{\omega} = \omega \mathbf{e}_\theta$ has only one non-zero component, and hence we set

$$\mathbf{\omega} := \frac{\partial v_r}{\partial z} - \frac{\partial v_z}{\partial r}. \quad (2.37)$$

From this definition and the continuity equation, we can derive two elliptic equations for the two non-zero velocity components. For radial velocity, we have

$$\frac{\partial^2 v_r}{\partial r^2} + \frac{\partial^2 v_r}{\partial z^2} = \frac{\partial \omega}{\partial z} - \frac{1}{r} \frac{\partial v_r}{\partial r} + \frac{1}{r^2} \frac{\partial}{\partial r} \left( \mathbf{v} \cdot \nabla \rho \right). \quad (2.38)$$

For axial velocity, we have

$$\frac{\partial^2 v_z}{\partial r^2} + \frac{\partial^2 v_z}{\partial z^2} = - \frac{\partial \omega}{\partial r} - \frac{1}{r} \frac{\partial v_r}{\partial z} - \frac{\partial}{\partial z} \left( \mathbf{v} \cdot \nabla \rho \right). \quad (2.39)$$

To find an equation for the vorticity, we take the curl of the momentum equations (2.33). This results in

$$\frac{\partial^2 (\mu \omega)}{\partial r^2} + \frac{\partial^2 (\mu \omega)}{\partial z^2} + \frac{\partial}{\partial r} \left( \mu \frac{\partial \omega}{\partial r} \right) + \frac{\partial}{\partial z} \left( \mu \frac{\partial \omega}{\partial z} \right) \quad (2.40)$$

In (2.40), the differential operator $\nabla$ is defined as

$$\nabla = \left( \frac{\partial}{\partial z}, 0, -\frac{\partial}{\partial r} \right)^T. \quad (2.41)$$

and $g = -ge_z$ with $g$ the gravitational acceleration equal to 9.82 m/s$^2$. The last equation that we will write in cylindrical coordinates is the energy equation in the form (2.34); it reads

$$c_p \left( \rho v_r \frac{\partial T}{\partial r} + \rho v_z \frac{\partial T}{\partial z} \right) = \frac{1}{r} \frac{\partial}{\partial r} \left( r \lambda \frac{\partial T}{\partial r} \right) + \frac{\partial}{\partial z} \left( \lambda \frac{\partial T}{\partial z} \right) - \sum_{i=1}^{N-1} h_i^* w_i. \quad (2.42)$$
2.3 Application problem: Bunsen flame

The combustion problem that we will focus on is the axisymmetric laminar Bunsen flame with one-step chemistry. This problem was previously presented by Bennett and Smooke in [Bennett and Smooke, 1998; Valdati, 1997], where it was used as an application problem for the local rectangular refinement (LRR) method. Results of simulations with the local defect correction method will be given in Chapter 7.

The physical configuration for this flame is shown in Figure 2.1. A mixture of methane and air flows up from a central jet. The jet is surrounded by a coflowing air stream. It is assumed that a steady conical flame has formed at the mouth of the cylindrical burner. The base of the flame is slightly wider than the diameter of the inner jet. As is indicated in the figure, the central jet has radius $r_1$. The thickness of the wall between the central jet and the coflowing jet equals $r_2 - r_1$. The surrounding jet has radius $r_3$.

In the chemical model we consider, there are five different chemical species ($N = 5$), namely methane ($\text{CH}_4$), oxygen ($\text{O}_2$), water ($\text{H}_2\text{O}$), carbon dioxide ($\text{CO}_2$), and the inert abundant species nitrogen ($\text{N}_2$). The single reaction describing the global chemical behavior of the gas mixture is

$$\text{CH}_4 + 2\text{O}_2 \rightarrow 2\text{H}_2\text{O} + \text{CO}_2.$$

(2.43)

We will choose the exponents $\gamma_j$ in (2.12) such that the reaction rates of the individual
Mathematical model for a stationary laminar flame

species equal

\[ w_{\text{CH}_4} = -W_{\text{CH}_4} w, \quad w_{\text{O}_2} = -2W_{\text{O}_2} w, \quad w_{\text{H}_2\text{O}} = 2W_{\text{H}_2\text{O}} w, \quad w_{\text{CO}_2} = W_{\text{CO}_2} w, \]

(2.44)

with

\[ w = k \frac{\rho Y_{\text{CH}_4}}{W_{\text{CH}_4}} \left( \frac{\rho Y_{\text{O}_2}}{W_{\text{O}_2}} \right)^2 = A \frac{\rho Y_{\text{CH}_4}}{W_{\text{CH}_4}} \left( \frac{\rho Y_{\text{O}_2}}{W_{\text{O}_2}} \right)^2 \exp \left( \frac{-E}{RT} \right). \]

(2.45)

Using (2.44), we can also elaborate the last term in the energy equation (2.42)

\[ \sum_{i=1}^{N-1} h_i^* w_i = (-h_{\text{CH}_4}^* W_{\text{CH}_4} - 2h_{\text{O}_2}^* W_{\text{O}_2} + 2h_{\text{H}_2\text{O}}^* W_{\text{H}_2\text{O}} + h_{\text{CO}_2}^* W_{\text{CO}_2}) w \]

\[ = \frac{h_{\text{CH}_4}^* W_{\text{CH}_4} + 2h_{\text{O}_2}^* W_{\text{O}_2} - 2h_{\text{H}_2\text{O}}^* W_{\text{H}_2\text{O}} - h_{\text{CO}_2}^* W_{\text{CO}_2}}{W_{\text{CH}_4}} \]

\[ = Q w_{\text{CH}_4}. \]

(2.46)

Here, Q is the heat release per unit mass of fuel consumed (dimension J/kg).

The parameter values we will use are taken from [Bennett and Smooke, 1998] Table 1, which are in turn from [Coffee et al., 1983]. These settings are based on an equivalence ratio of \( \Phi = 0.776 \). The values have been converted to standard units and are listed in Table 2.1. It should be noted that the Arrhenius reaction rate pre-exponential factor in [Bennett and Smooke, 1998] is different from ours in (2.41). If we denote the factor from [Bennett and Smooke, 1998] by \( \tilde{A} \), the two are related via

\[ A = \tilde{A} W_{\text{O}_2}^2. \]

(2.47)

The source terms in the species continuity equations and in the energy equation in [Bennett and Smooke, 1998] are expressed in the auxiliary variable \( q \). The variable \( q \) relates to \( w \) from (2.45) in the following way

\[ q := Q W_{\text{CH}_4} w. \]

(2.48)

There is a misprint in the definition of \( q \) in [Bennett and Smooke, 1998] p. 226; multiplication by the heat release \( Q \) should be added at the right hand side. This can be verified by checking the dimensions of the individual terms in e.g. the energy equation. The mass density \( \rho \) is determined from the equation of state (2.25), in which the pressure \( p \) is assumed to be constant and equal to the atmospheric pressure \( p_{\text{atm}} = 1.0133 \cdot 10^5 \) N/m². Next, the prescribed \( \rho \lambda \) is used to find the thermal conductivity \( \lambda \). Finally, we employ the definition of the Prandtl number, viz.

\[ \text{Pr} := \frac{\mu c_p}{\lambda}, \]

(2.49)

to find the value of the viscosity \( \mu \).

As the flame is axisymmetric, the problem can be formulated as a two-dimensional boundary value problem in cylindrical coordinates. The computational domain extends from \( r = 0.0 \) cm to \( r = 3.0 \) cm in radial direction and from \( z = 0.0 \) cm to \( z = 25 \) cm in axial direction. Boundary conditions have to be provided for the four domain boundaries.
### 2.3 Application problem: Bunsen flame

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td>$8.3456 \cdot 10^8 \text{ m}^6/(\text{mol}^2 \text{s})$</td>
</tr>
<tr>
<td>$E$</td>
<td>$1.2594 \cdot 10^5 \text{ J/mol}$</td>
</tr>
<tr>
<td>$Q$</td>
<td>$4.9823 \cdot 10^7 \text{ J/kg}$</td>
</tr>
<tr>
<td>$c_p$</td>
<td>$1.2887 \cdot 10^3 \text{ J/(kg K)}$</td>
</tr>
<tr>
<td>$\rho \lambda$</td>
<td>$2.3681 \cdot 10^{-2} \text{ kg J/(m}^4 \text{s K})$</td>
</tr>
<tr>
<td>$Pr$</td>
<td>0.7</td>
</tr>
</tbody>
</table>

**Table 2.1**: Parameter values for the Bunsen flame.

At the *inflow boundary*, we assume that the velocity profile has reached a parabolic profile across the inner jet. The velocity vanishes across the thickness of the wall, and it increases exponentially across the outer jet. To prevent both *flash-back* and *blow-off* of the flame, the average velocity at the inner jet (equal to $v_{z,\text{inner}}$) should be in the range of $25 \text{ cm/s}$ to $62.5 \text{ cm/s}$, see [Harris et al., 1949]. Following [Bennett and Smooke, 1998 Sec 2.1] and [Valdati, 1997 Sec. 2.3.1], we set $v_{z,\text{inner}} = 40 \text{ cm/s}$. The maximum velocity of the coflow at the outer radial boundary is $v_{z,\text{outer}} = 40 \text{ cm/s}$. For the vorticity boundary condition, we use (2.37). In the numerical simulations, the derivatives in (2.37) will be discretized. For the derivative of the radial velocity, a one-sided difference will be used. Note that this implies that the discrete boundary condition depends on the solution inside the domain. Temperature is set to $298 \text{ K}$ along the inflow boundary. For the individual species, mass is conserved. This results in the following boundary conditions at the inflow boundary:

\[
v_r = 0, \quad v_z = \begin{cases} 
2v_{z,\text{inner}} \left(1 - \frac{(r/r_1)^2}{2}\right), & 0 \leq r \leq r_1, \\
0, & r_1 \leq r \leq r_2, \\
v_{z,\text{outer}} \left(1 - \exp\left(-\frac{(r - r_2)}{w_{BL}}\right)\right), & r_2 \leq r \leq r_3,
\end{cases}
\]  

(2.50)

with $r_1 = 0.50 \text{ cm}$, $r_2 = 0.55 \text{ cm}$, $r_3 = 3.0 \text{ cm}$, and $w_{BL} = 0.20 \text{ cm}$. Further,

\[
\omega = \frac{\partial v_r}{\partial z} - \frac{\partial v_z}{\partial r}, \quad T = 298 \text{ K,}
\]  

(2.51)

and ($i = 1, 2, \ldots, N$)

\[
\begin{align*}
\rho v_z (Y_i - Y_{\text{inner},i}) &= \frac{1}{\text{Le}_i} \frac{\lambda}{c_p} \frac{\partial Y_i}{\partial z}, \quad 0 \leq r \leq r_2, \\
\rho v_z (Y_i - Y_{\text{outer},i}) &= \frac{1}{\text{Le}_i} \frac{\lambda}{c_p} \frac{\partial Y_i}{\partial z}, \quad r_2 \leq r \leq r_3,
\end{align*}
\]  

(2.52)

The Lewis number $\text{Le}_i$ and the mass fractions $Y_{\text{inner},i}$ and $Y_{\text{outer},i}$ are given in Table 2.2. Boundary condition (2.52) can be obtained by integration of the species continuity equation (2.35) over a volume with its top boundary aligned with the inflow boundary.

At the *axis of symmetry* (the $z$-axis), we use ($i = 1, 2, \ldots, N$)

\[
v_r = 0, \quad \frac{\partial v_z}{\partial r} = 0, \quad \omega = 0, \quad \frac{\partial T}{\partial r} = 0, \quad \frac{\partial Y_i}{\partial r} = 0.
\]  

(2.53)
At the outer radial boundary, we impose
\[ \frac{\partial}{\partial r}(rv_r) = 0, \quad v_z = v_{z,\text{outer}} \left( 1 - \exp \left( -(r_3 - r_2)/w_{\text{BL}} \right) \right), \quad (2.54) \]
and (i = 1, 2, ... , N)
\[ \omega = \frac{\partial v_r}{\partial z} - \frac{\partial v_z}{\partial r}, \quad \frac{\partial T}{\partial r} = 0, \quad \frac{\partial Y_i}{\partial r} = 0. \quad (2.55) \]
The boundary condition for the radial velocity in (2.54) may be found from the continuity equation, if we assume that the density is constant. In the numerical simulations, the vorticity boundary condition is treated in a similar way as described for the boundary condition at the inflow boundary.

The outflow boundary is assumed to be located sufficiently far away from the flame. Hence, we apply the boundary conditions (i = 1, 2, ... , N)
\[ v_r = 0, \quad \frac{\partial v_z}{\partial z} = 0, \quad \frac{\partial \omega}{\partial z} = 0, \quad \frac{\partial T}{\partial z} = 0, \quad \frac{\partial Y_i}{\partial z} = 0, \quad (2.56) \]
at the outflow boundary.

Bennett and Smooke [Bennett and Smooke, 1998; Valdati, 1997] have shown that all but one of the dependent variables in the Bunsen flame problem have large gradients in a very small region of the computational domain. For this reason, adaptive gridding is a must for this simulation. Numerical results for the problem formulated in this section will be presented in Chapter 7.

Table 2.2: Lewis numbers and mass fractions at the inflow boundary.

<table>
<thead>
<tr>
<th>CH₄</th>
<th>O₂</th>
<th>H₂O</th>
<th>CO₂</th>
<th>N₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>Le</td>
<td>0.96</td>
<td>1.1</td>
<td>0.83</td>
<td>1.39</td>
</tr>
<tr>
<td>Y_{inner}</td>
<td>0.04305</td>
<td>0.22201</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Y_{outer}</td>
<td>0</td>
<td>0.23200</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Chapter 3

A two-grid local defect correction method

3.1 An introductory example

In this section, we will explain the basic principle of the LDC algorithm. To this end, we consider a very simple one-dimensional model problem, viz.

\[
\begin{align*}
  u'' &= f, \quad \text{in } \Omega = (0, 1), \\
  u(0) &= u_0, \quad u(1) = u_1.
\end{align*}
\]  

(3.1)

In (3.1), the source term \( f \) and boundary values \( u_0 \) and \( u_1 \) are chosen such that

\[
u(x) = \frac{1}{2} \left[ \tanh(50(x - 1/8)) + 1 \right].
\]  

(3.2)

The continuous solution of boundary value problem (3.1) has a high activity area around \( x = 1/8 \) and is smooth in the rest of the domain. Due to the activity, a grid with a high resolution is needed to represent \( u \) in the area around \( x = 1/8 \), whereas a grid with far less resolution is needed in the rest of the domain. For this reason, we will use a **global coarse grid** to discretize (3.1) and we will use a **local fine grid** in the area \( \Omega_l = (0, \gamma) \) with \( \gamma = 3/10 \). The latter value is such that \( \gamma \) is a point of the coarse grid. For the global coarse grid, we define \( H = 1/20 \), \( N = 1/H \), and \( x_i := iH \), \( i = 0, 1, \ldots, N \), and set

\[
\Omega^H := \{ x_i \mid i = 1, 2, \ldots, N - 1 \},
\]  

(3.3)

For the local fine grid in \( \Omega_l \), we use

\[
\Omega^h := \{ ih \mid i = 1, 2, \ldots, n - 1 \},
\]  

(3.4)

in which \( h := H/4 = 1/80 \), \( n := \gamma/h \). For these settings, the LDC algorithm can be formulated as follows. The first initializing step is to discretize (3.1) on the global coarse
grid. Using the standard central difference discretization, we find the following system of equations
\[
\begin{aligned}
\frac{u^H_{i+1}(x_i) - 2u^H_i(x_i) + u^H_{i-1}(x_i)}{H^2} &= f(x_i), \quad i = 1, 2, \ldots, N - 1, \\
u^H_0(0) &= u_0, \quad u^H_1(1) = u_1.
\end{aligned}
\] (3.5)

The initial coarse grid approximation \(u^H_1\) can be used to define a new boundary value problem on \(\Omega_1\): we use the coarse grid approximation at \(x = \gamma\) to set
\[
\begin{aligned}
u'' &= f, \quad \text{in } \Omega_1 = (0, \gamma), \\
u(0) &= u_0, \quad u(\gamma) = u^H_0(\gamma).
\end{aligned}
\] (3.6)

Discretization of (3.6) with central differences on the local fine grid \(\Omega_1^h\) yields an approximation \(u_{i,0}^h\), which is assumed to be more accurate than the coarse grid approximation \(u^H_0\). For grid points of the global coarse grid that lie within \(\Omega_i\), we have found two approximations. The local fine grid approximation enables us to give an estimate of the local discretization error, which in turn can be used to improve the coarse grid approximation. The local discretization error of the coarse grid discretization (3.5) is defined as \((i = 1, 2, \ldots, N - 1)\)
\[
d(x_i) := \frac{u(x_{i+1}) - 2u(x_i) + u(x_{i-1})}{H^2} - f(x_i).
\] (3.7)

If we would know the values \(d(x_i), i = 1, 2, \ldots, N - 1\), then we could add these values to the right hand side of the first equation in (3.5), and solve this modified system of equations to find the exact solution of boundary value problem (3.7). Note however that we have used the continuous solution \(u\) in the definition (3.7) of the local discretization error. Since \(u\) is normally unknown, we cannot calculate \(d(x_i)\). What we can do though, is use the approximation \(u^h_{i,0}\) calculated on the local fine grid to estimate \(d(x_i)\) for those grid points of the global coarse grid that lie within \(\Omega_i\). If \(k := \gamma/H\), then we have
\[(i = 1, 2, \ldots, k - 1)\]
\[
d(x_i) \approx \frac{u_{i,0}^b(x_{i+1}) - 2u_{i,0}^b(x_i) + u_{i,0}^b(x_{i-1})}{H^2} - f(x_i) =: d^h_{i,0}(x_i).
\] (3.8)

If we set \(d^h_{i,0}(x_i) := 0\) for \(i = k, k + 1, \ldots, N - 1\), then we can formulate a new discrete problem on the global coarse grid by placing the estimate for the local discretization error in the right hand side of the first equation of (3.5), viz.
\[
\begin{aligned}
\frac{u^H_{i+1}(x_i) - 2u^H_i(x_i) + u^H_{i-1}(x_i)}{H^2} &= f(x_i) + d^h_{i,0}(x_i), \quad i = 1, 2, \ldots, N - 1, \\
u^H_0(0) &= u_0, \quad u^H_1(1) = u_1.
\end{aligned}
\] (3.9)

The system (3.9) yields an updated coarse grid approximation \(u^H_1\), which is assumed to be more accurate than \(u^H_0\). In particular, we have found a new value \(u^H_1(\gamma)\) at \(x = \gamma\), that we can use to define a new local boundary value problem similar to (3.6). Discretization of this problem on the local fine grid \(\Omega_1^h\) leads to a new approximation \(u_{i,1}^h\) on the
Figure 3.1: Numerical results of the LDC algorithm applied to boundary value problem (3.1). The solid curve in each figure is the continuous solution $u$. 
local fine grid. Figure 3.1 shows the numerical results of this procedure for the simple
problem considered here.

One can show that for this one-dimensional example problem, the LDC algorithm has
reached a fixed point after only one correction step. Therefore, more iterations would
produce the same approximations as shown in Figures 3.1(c) and (d). In general, this is
not true. For other cases, the LDC method shows fast convergence, but does not reach a
fixed point after only one iteration. We will study the convergence behavior of the LDC
method in Chapter 5.

3.2 Formulation of the LDC method

To formulate the LDC algorithm more generally and for two-dimensional problems, we
consider the elliptic boundary value problem

\[
\begin{align*}
Lu &= f, & \text{in } \Omega, \\
u &= g, & \text{on } \partial \Omega.
\end{align*}
\] (3.10)

In (3.10), L is a linear elliptic differential operator, and f and g are the source term and
Dirichlet boundary condition, respectively. To discretize (3.10), we first choose a global
coarse grid (grid size \(H\)), which we denote by \(\Omega^H\). An initial approximation \(u^H_0\) on \(\Omega^H\)
can be found by solving the system

\[
L^H u^H_0 = f^H,
\] (3.11)

which is a discretization of boundary value problem (3.10). In (3.11), 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 them as \(L = \Delta\), and \(L^H\)
the standard five-point stencil approximating \(\Delta\). Here, \(\Delta\) denotes the two-dimensional
Laplacian operator.

Now, assume that the continuous solution \(u\) of (3.10) has a high activity region in some
(small) part of the domain. This high activity of \(u\) may be captured by discretizing (3.10)
on a composite grid. To this end, we choose \(\Omega_1 \subset \Omega\) such that the high activity region
of \(u\) is contained in \(\Omega_1\). In \(\Omega_1\), we choose a local fine grid (grid size \(h\)), which we denote
by \(\Omega^h_1\). The fine grid is chosen such that \(\Omega^H \cap \Omega_1 \subset \Omega^h_1\), i.e., grid points of the global
coarse grid that lie in the area of refinement belong to the local fine grid too.

In order to formulate a discrete problem on \(\Omega^h_1\), we have to define artificial boundary con-
ditions on \(\Gamma\), the interface between \(\Omega_1\) and \(\Omega \setminus \Omega_1\), see Figure 3.2. We prescribe artificial
Dirichlet boundary conditions by applying an interpolation operator \(P^{h,H}\). The oper-
ator \(P^{h,H}\) maps function values at grid points of the coarse grid that lie on the interface,
denoted by \(\Gamma^H\), to function values at grid points of the fine grid that lie on the interface,
denoted by \(\Gamma^h\). If we denote the vector space of grid functions on \(\Gamma^H\) by \(G(\Gamma^H)\), and likewise introduce \(G(\Gamma^h)\), we have \(P^{h,H} : G(\Gamma^H) \to G(\Gamma^h)\). In practice, we take \(P^{h,H}\) to be the linear interpolation operator on the interface.
3.2 Formulation of the LDC method

In this way, we find the following approximation $u_{h,i}^{\Omega_l}$, $i = 0$, on $\Omega_l^h$:

$$L_{h,l}^h u_{h,i}^{\Omega_l} = f_{h,l}^i - B_{h,l}^h \mathcal{P}_{h,H}^i (u_H^{\mid \Gamma}).$$

(3.12)

In (3.12), the matrix $L_{h,l}^h$ is a discrete approximation of the differential operator $L$ on the subdomain $\Omega_l$, and the first term on the right hand side $f_{h,l}^i$ incorporates the source term $f$ as well as the Dirichlet boundary condition $g$ on $\partial \Omega_l \setminus \Gamma$ given in (3.10). The dependence on the coarse grid approximation via the artificial Dirichlet boundary condition is made explicit by the second term on the right hand side. We assume $L_{h,l}^h$ to be invertible.

When boundary value problem (3.10) has been discretized and solved on a coarse grid, and when an area of the coarse grid has been refined and a local solution has been calculated on the finer grid, we can use the local fine grid solution to update the coarse grid approximation. This update can be achieved by projecting the more accurate fine grid solution onto the local coarse grid, and by calculating the residual of the projected solution; this residual is an estimate for the local discretization error of the coarse grid discretization. The estimate is used to formulate a modified discrete problem on the coarse base grid. This is considered in detail in what follows.

The grid points of the coarse grid will be partitioned as follows

$$\Omega^H = \Omega_l^H \cup \Gamma^H \cup \Omega_c^H,$$

(3.13)

where $\Omega_l^H := \Omega_l \cap \Omega$, $\Gamma^H := \Omega^H \cap \Gamma$ and $\Omega_c^H := \Omega^H \setminus (\Omega_l^H \cup \Gamma^H)$. See Figure 3.3. Using
A two-grid local defect correction method

Figure 3.3: A global coarse grid and its partitioning. The darker area is the area of high activity $\Omega_1$. The interface $\Gamma$ is dashed. We have $\Omega^H = \{1\}$, $\Gamma^H = \{2,4,5\}$, and $\Omega^H = \{3,6,7,8,9\}$. Grid points on the boundary are labeled $D$.

this partitioning of $\Omega^H$, we set

$$u^H := \begin{pmatrix} u^H_l \\ u^H_{\Gamma} \\ u^H_c \end{pmatrix}.$$  \hspace{1cm} (3.14)

We will apply this same partitioning for other grid functions in $G(\Omega^H)$. Assuming that the stencil at grid point $(x,y)$ involves (at most) function values at $(x+iH, y+jH)$ with $i,j \in \{-1,0,1\}$, we can partition the discrete operator $L^H$ as

$$L^H = \begin{pmatrix} L^H_l & B^H_{l,\Gamma} & 0 \\ B^H_{l,\Gamma} & L^H_{\Gamma} & B^H_{\Gamma,c} \\ 0 & B^H_{c,\Gamma} & L^H_c \end{pmatrix}.$$  \hspace{1cm} (3.15)

Using this decomposition, the coarse grid initialization (3.11) may also be written as

$$\begin{pmatrix} L^H_l & B^H_{l,\Gamma} & 0 \\ B^H_{l,\Gamma} & L^H_{\Gamma} & B^H_{\Gamma,c} \\ 0 & B^H_{c,\Gamma} & L^H_c \end{pmatrix} \begin{pmatrix} u^H_l,0 \\ u^H_{\Gamma,0} \\ u^H_{c,0} \end{pmatrix} = \begin{pmatrix} f^H_l \\ f^H_{\Gamma} \\ f^H_c \end{pmatrix}.$$  \hspace{1cm} (3.16)

After one coarse grid solve and one local fine grid solve, we have found the coarse grid approximations $u^H_l,0$, $u^H_{\Gamma,0}$ and $u^H_{c,0}$, and the fine grid approximation $u^h_l,0$. We have one approximation at each grid point $(x,y)$ of $\Gamma^H$ and $\Omega^H$, namely $u^H_l,0(x,y)$ and $u^H_{c,0}(x,y)$, respectively, and we have two approximations at each grid point $(x,y)$ of $\Omega^H$, namely both $u^H_l,0(x,y)$ and $u^H_{c,0}(x,y)$. Of these last two approximations, $u^H_{c,0}(x,y)$ is considered more accurate, as it is calculated on a finer grid. We try to improve the coarse grid approximation by using the approximation $u^H_{c,0}$ calculated on the local fine grid to estimate the local discretization error of the coarse grid discretization. For the description
of this step, we introduce the operator \( R^{H,h} : G(\Omega^h) \rightarrow G(\Omega^H) \) as the restriction from \( \Omega^h \) onto \( \Omega^H \), viz. \( (R^{H,h} u^h_i(x,y) = u^h_i(x,y) \) for all \( u^h_i \in G(\Omega^h), (x,y) \in \Omega^H \).

If we would substitute the projection on \( \Omega^H \) of the exact solution \( u \) of boundary value problem (3.10) into the coarse grid discretization (3.11), we would find the local discretization error or defect \( d^H \), given by

\[
L^H(u|\Omega^H) = f^H + d^H.
\]

In particular, we would find the following local defect on \( \Omega^H \):

\[
L^H(u|\Omega^H) + B^H_{i,r}(u_{i,r}) = f^H + d^H.
\] (3.17)

If we would know the values of the defect \( d^H \), we could use them to find a better approximation on the coarse grid. This could be achieved by putting the defect vector on the right hand side of (3.11). However, as we do not know the exact solution of the boundary value problem, we can calculate neither \( d^H \) nor \( d^H \). What we can do though, is to use the approximation \( u_{i,0}^H \) calculated on the local fine grid to estimate \( d^H \). Using (3.17), we find

\[
d^H_i = L^H_i(u|\Omega^H) + B^H_{i,r}(u_{i,r}) - f^H_i \approx L^H_iR^{H,h}u^h_{i,0} + B^H_i u^H_{i,0} - f^H_i =: d^H_{i,0}.
\] (3.18)

Using (3.18), we find an estimate of the local discretization error of the coarse grid discretization at all points of \( \Omega^H \). Therefore, we can update the coarse grid approximation by placing the estimate (3.18) at the right hand side of the coarse grid equation (3.11) or (3.16). This leads to the coarse grid correction step to find \( u^H_i, i = 1 \), on the coarse grid

\[
L^H u^H_i = \begin{pmatrix} f^H_i + d^H_{i,i-1} \\ f^H_i \\ f^H_i \end{pmatrix} = \begin{pmatrix} L^H_i R^{H,h} u^h_{i,i-1} + B^H_i u^H_{i,1} - f^H_i \\ f^H_i \\ f^H_i \end{pmatrix}.
\] (3.19)

Previous results [Anthonissen et al., 2000; Ferket, 1995; Hackbusch, 1984; Wappler, 1999] show, that it may be beneficial to use the estimate (3.18) not at all points of \( \Omega^H \), but in a subset of \( \Omega^H \) only. In particular, nodes in \( \Omega^H \) lying close to the interface \( \Gamma \) should be excluded. This leads to the introduction of what might be called a safety region, see Figure 3.4, in which a subset \( \Omega^H_{def} \) of \( \Omega^H \) has been chosen. The estimate for the local discretization error of the coarse grid discretization is placed on the right hand side of equations corresponding to grid points belonging to the set \( \Omega^H_{def} \) only. This leads to the following generalization of (3.19):

\[
L^H u^H_i = \begin{pmatrix} f^H_i + X^H_i d^H_{i,i-1} \\ f^H_i \\ f^H_i \end{pmatrix} = \begin{pmatrix} (I - X^H_i) f^H_i + X^H_i (L^H_i R^{H,h} u^h_{i,i-1} + B^H_i u^H_{i,1}) \\ f^H_i \\ f^H_i \end{pmatrix}.
\] (3.20)

In (3.20), we have used \( X^H_i : G(\Omega^H_i) \rightarrow G(\Omega^H) \) which is defined by

\[
(X^H_i u^H_i)(x,y) := \begin{cases} u^H_i(x,y), & (x,y) \in \Omega^H_{def} \\ 0, & (x,y) \in \Omega^H \setminus \Omega^H_{def} \end{cases}
\] (3.21)
A two-grid local defect correction method

Figure 3.4: The area of refinement is the union of the light and dark gray areas. The safety region is light gray. The defect is only calculated in grid points lying inside the dark gray area. For this situation, the sets \( \Omega_H^L \) and \( \Omega_H^{\text{def}} \) consist of nine and four grid points, respectively.

for all grid functions \( u_H^1 \in G(\Omega_H^H) \). Note that \( X_H^1 \) is the matrix corresponding to multiplication of a grid function on \( \Omega_H^H \) by the characteristic function of \( \Omega_H^{\text{def}} \). If \( \Omega_H^{\text{def}} = \Omega_H^H \), we use the estimate for the discretization error at all points of \( \Omega_H^H \), and (3.20) reduces to (3.19). If \( \Omega_H^{\text{def}} = \emptyset \), we do not use any components of the estimate of the discretization error, and (3.20) reduces to the initial system (3.11) or (3.16). Therefore, the situation \( \Omega_H^{\text{def}} = \emptyset \) yields \( u_H^1 = u_H^0 \), and is not of interest; in what follows, we will assume \( \Omega_H^{\text{def}} \neq \emptyset \). Under this assumption, (3.20) produces a new solution \( u_H^1 \) on the coarse grid. Because (3.20) incorporates estimates of the local discretization error of the coarse grid discretization, the new solution \( u_H^1 \) is assumed to be more accurate than \( u_H^0 \). Hence, the new solution \( u_H^1 \) provides a better boundary condition on the interface \( \Gamma \). A better solution on the local fine grid can be found as before by solving (3.12) with \( i = 1 \).

Summarizing, we have found the following iterative method.

Algorithm 3.1
Two-grid LDC algorithm with a priori chosen area of refinement

Initialization

- Solve the basic coarse grid problem (3.11).
- Solve the local fine grid problem (3.12) with \( i = 0 \).

Iteration, \( i = 1, 2, \ldots \)

- Solve the updated coarse grid problem (3.20).
- Solve the local fine grid problem (3.12).
### 3.3 Some properties

The following lemma shows that once the coarse grid approximations do not change value on the interface, the LDC algorithm converges and a fixed point of the iteration has been reached.

**Lemma 3.2** If \( u^H_{i,k} = u^H_{i,k-1} \) for a certain index \( k \), then Algorithm 3.1 converges and it has reached a fixed point, i.e.,

\[
\begin{align*}
  u_i^H &= u_k^H, \\
  u_i^h &= u_{i,k}^h,
\end{align*}
\]

(3.22)

for all \( i = k, k + 1, \ldots \)

**Proof.** Assume that \( u^H_{i,k} = u^H_{i,k-1} \) for a certain index \( k \). From (3.12), we have that \( u_i^h = u_i^h_{k-1} \), and hence, from (3.20),

\[
L^H u_{k+1}^H = \begin{pmatrix}
(I - X_i^H) f_i^H + X_i^H (L_i^H R_i^H u_i^h_{k-1} + B_i^H u_i^H) \\
\end{pmatrix}
\]

(3.23)

Because we have assumed \( L^H \) to be invertible, we have \( u^H_{k+1} = u^H_k \) for all grid points in the global coarse grid. As \( f^H \subset \Omega^H \), we have \( u^H_{k,k+1} = u^H_{k,k} \). By induction, we find \( u^H_i = u_i^k \) and \( u_i^h = u_i^k \) for all \( i = k, k + 1, \ldots \)

The equations describing the iteration, (3.12), (3.20), can be written in matrix notation as

\[
\begin{pmatrix}
L_i^H & 0 & B_{i,l}^h & 0 \\
0 & L_i^H & B_{i,r}^h & 0 \\
0 & B_{i,l}^h & L_i^H & B_{i,c}^h \\
0 & 0 & B_{i,c}^h & L_i^c^H \\
\end{pmatrix}
\begin{pmatrix}
u_i^h \\
u_i^l \\
u_i^h \\
u_i^c \\
\end{pmatrix}
= \begin{pmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\end{pmatrix}
\begin{pmatrix}
u_{i-1}^h \\
u_{i-1}^l \\
u_{i-1}^h \\
u_{i-1}^c \\
\end{pmatrix}
+ \begin{pmatrix}
f_i^h \\
f_i^l \\
f_i^h \\
f_i^c \\
\end{pmatrix}.
\]

(3.23)

We will use the following short notation for (3.23):

\[
L_{i,h}^H u_{i,h}^H = S_{i,h}^H u_{i-1,h}^H + f_{i,h}^H.
\]

(3.24)

If the LDC algorithm converges, then this equation has a fixed point, which we will denote by

\[
u_{i,h}^H = \begin{pmatrix}
u_i^h \\
u_i^l \\
u_i^h \\
u_i^c
\end{pmatrix},
\]

(3.25)
where the same partitioning is used as above. The fixed point \( u^{H,h} \) satisfies, by definition,
\[
L^{H,h}u^{H,h} = S^{H,h}u^{H,h} + ³^{H,h}.
\]

(3.26)

For the LDC algorithm without a safety region, the fixed point \( u^{H,h} \) has the nice properties stated in the next theorem. Theorem 3.3 corresponds with the results from [Ferket, 1996, Sec. 3.2]. Equation (3.28) forms the composite grid discretization given by the LDC algorithm.

**Theorem 3.3** Consider the LDC iteration for the special case that there is no safety region, i.e., \( \Omega^H_{def} = \Omega^H, \ X^H = I \). Assume that the LDC iteration converges to a fixed point \( u^{H,h} \). Then \( u^{H,h} \) has the following two properties:

- the projection of \( u^h \) on the local coarse grid equals \( u^H \), viz.
  \[
  R^{H,h}u^h = u^H,
  \]

(3.27)

- \( u^h, u^H, \) and \( u^c \) satisfy the system of equations
  \[
  \begin{pmatrix}
  L^h & B^h_{l,r}p^h & 0 \\
  B^H_{l,r}R^H,h & L^H & B^H_{l,c} \\
  0 & B^H_{c,l} & L^H_c
  \end{pmatrix}
  \begin{pmatrix}
  u^h \\
  u^H \\
  u^c
  \end{pmatrix}
  =
  \begin{pmatrix}
  f^h \\
  f^H \\
  f^c
  \end{pmatrix}.
  \]

(3.28)

**Proof.** From (3.26), we find
\[
(L^{H,h} - S^{H,h})u^{H,h} = ³^{H,h}.
\]

Substituting the matrices \( L^{H,h} \) and \( S^{H,h} \) and using \( X^H = I \) gives
\[
\begin{pmatrix}
  L^h & B^h_{l,r}p^h & 0 \\
  B^H_{l,r}R^H,h & L^H & B^H_{l,c} \\
  0 & B^H_{c,l} & L^H_c
  \end{pmatrix}
  \begin{pmatrix}
  u^h \\
  u^H \\
  u^c
  \end{pmatrix}
  =
  \begin{pmatrix}
  f^h \\
  f^H \\
  f^c
  \end{pmatrix}.
  \]

(3.29)

The second equation of this system reads
\[
-L^H R^H,h u^h + L^H u^H = 0,
\]

which gives the first claim. Elimination of \( u^h \) from (3.29) gives (3.28). □

Ferket [Ferket, 1996] has studied discretization errors of the composite grid discretization implicitly given by the LDC algorithm for a model problem. He considers Poisson’s equation on the unit square with zero Dirichlet boundary conditions, viz.
\[
\begin{cases}
  \Delta u = f, & \text{in } \Omega = (0,1)^2, \\
  u = 0, & \text{on } \partial \Omega.
\end{cases}
\]

(3.30)

The area of refinement is located at the lower left corner of \( \Omega \):
\[
\Omega_l = (0,\gamma_1) \times (0,\gamma_2).
\]

(3.31)
3.3 Some properties

Ferket forms the composite grid matrix, [Ferket, 1996 form. (4.13)], and proves a stability result for it [Ferket, 1996, Theorem 4.19] as well as a strong damping of errors in fine grid points located near the interface [Ferket, 1996 Theorem 4.20]. These results lead to an upper bound for the global discretization error [Ferket, 1996 Theorem 4.22]; this important theorem is as follows.

**Theorem 3.4** Consider the LDC algorithm applied to the model problem (3.30). Let \( \Omega_1 \) be defined as in (3.31). Assume that there is no safety region. Then the fixed point \( u^{H,h} \) of the iteration satisfies

\[
\|u - u^{H,h}\|_\infty \leq \frac{13}{8} \max \{C_1, C_2\} h^2 + \frac{1}{8} C_3 H^2 + 3D_1 H^j.
\]  

(3.32)

The exponent \( j \) in (3.32) is 1 if piecewise linear interpolation is used on the interface; it is 2 if piecewise quadratic interpolation is used. The constants \( C_1, C_2, C_3, \) and \( D_1 \) are defined by

\[
C_1 := c_1 \max \left\{ |u^{(4)}(x,y)| : (x,y) \in (0, \gamma_1 - h) \times (0, \gamma_2 - h) \right\}, \tag{3.33}
\]

\[
C_2 := c_2 \max \left\{ |u^{(4)}(x,y)| : (x,y) \in \Omega_1 \setminus ((0, \gamma_1 - 2h) \times (0, \gamma_2 - 2h)) \right\}, \tag{3.34}
\]

\[
C_3 := c_3 \max \left\{ |u^{(4)}(x,y)| : (x,y) \in \Omega \setminus ((0, \gamma_1 - H) \times (0, \gamma_2 - H)) \right\}, \tag{3.35}
\]

\[
D_1 := d_1 \max \left\{ |u^{(1+1)}(x,y)| : (x,y) \in \Gamma \right\}, \tag{3.36}
\]

in which \( c_1, c_2, c_3, \) and \( d_1 \) are independent of \( H, h, \) and \( u.\)

The error bound in Theorem 3.4 separates the contributions to the error by the discretization error in the high activity area (first term of the bound), the error in the low activity area (second term), and the error due to interpolation on the interface (third term). In (3.32), the constants \( C_1, C_2, \) and \( C_3 \) depend on the values of fourth derivatives of the continuous solution \( u \) in the high activity area, near the interface, and outside the high activity area, respectively. The constant \( D_1 \) depends on second derivatives if piecewise linear interpolation is used on the interface; it depends on third derivatives if piecewise quadratic interpolation is used.

The constants \( C_1, C_2, \) and \( C_3 \) measure the smoothness of the continuous solution \( u \) of the boundary value problem. The local discretization error of the coarse grid discretization at a grid point \( (x,y) \in \Omega_1^H \) is approximately \( C_1 H^2 \). Note that \( C_1 \) is large as we have assumed that \( u \) has a high activity area in \( \Omega_1 \). Because \( C_1 \) is determined by the smoothness of the continuous solution, the only way to reduce the local discretization error is to reduce the grid size \( H \) of the coarse grid. Using LDC, the local discretization error at the same grid point \( (x,y) \) is approximately \( C_1 h^2 \), with \( h \) the grid size of the local fine grid. Hence this error is already four times smaller when \( h = H/2 \). This effect is reflected in the first term at the right hand side of (3.32), where the constants \( C_1 \) and \( C_2 \) are multiplied by \( h^2 \).

The sharpness of the bound in Theorem 3.4 is studied in some numerical experiments in [Ferket, 1996 Sec 4.4]. The third term at the right hand side of (3.32) is \( O(H) \) if piecewise linear interpolation is used on the interface. This linear dependence is shown to
hold for a model problem in which all other error terms vanish (i.e., $C_1 = C_2 = C_3 = 0$). In a second model problem with a less smooth solution, the values of the discretization errors found for linear and quadratic interpolation on the interface are close together. The reason is that the first two error terms at the right hand side of (3.32) (related to discretization errors) dominate the third term (related to the interpolation error). Based on these observations, we will employ linear interpolation rather than quadratic interpolation at the interface in the numerical experiments in this thesis.
Chapter 4

An LDC method for finite volume discretizations

In this chapter, we consider the integral formulation of a two-dimensional convection-diffusion problem. We combine the LDC technique with standard finite volume discretizations of this problem on the global coarse and local fine grids. In a straightforward generalization of the LDC algorithm as presented in Chapter 3, the discrete conservation property, which is one of the main attractive features of a finite volume method, does not hold for the composite grid approximation. Here, we present a modified LDC method, which is based on a special form of the defect correction term used in the right hand side of the coarse grid problem. Due to this finite volume adapted defect correction term, the conservation property is preserved in the discretization on the composite grid. As we will prove, the conservation property on the composite grid holds for the limit LDC discretization, i.e., for the number of LDC iterations tending to infinity. In practice, however, for efficiency reasons one applies only one or two LDC iterations. In numerical experiments we observe that, due to the very high convergence rate of the LDC iteration (cf. [Ferket, 1996, Ferket and Reusken, 1996a, Ferket and Reusken, 1996b, Hackbusch, 1984]), the result after one or two LDC iterations is in general already satisfactory (with respect to conservation).

Attractive features of the finite volume based LDC method presented here are:

- the method yields a discretization on locally refined grids;
- a discrete conservation property holds for the discretization on the composite grid;
- the method is simple: it only uses standard finite volume discretizations on uniform (global coarse and local fine) grids.

In Section 4.1, we formulate a stationary convection-diffusion problem, and discuss a standard vertex-centered finite volume technique for discretizing this problem on a uni-
form grid. In Section 4.2, we briefly recall the concept of composite grids, and derive a finite volume adapted LDC method. For the resulting composite grid discretization, we prove a discrete conservation property. In Section 4.3, we show results of a few numerical experiments.

The generalization of the LDC method to finite volume discretizations studied in this chapter was previously presented in [Anthonissen et al., 1998, Anthonissen et al., 1999] and [van ‘t Hof, 1998, Ch. 7].

4.1 Problem formulation and finite volume discretization on a uniform grid

We consider a stationary convection-diffusion problem in the domain \( \Omega = (0,1) \times (0,1) \) with Dirichlet boundary conditions \( \varphi = \psi \) on \( \partial \Omega \). By \( V \subset \Omega \) we denote a generic subdomain of \( \Omega \). The outward unit normal vector to \( \partial V \) is denoted by \( n \). We assume given functions \( D = D(x,y) \geq D_{\text{min}} > 0 \) (diffusion coefficient), \( v = (v_1(x,y), v_2(x,y))^T \) (mass flux), and \( s = s(x,y) \) (source term). Introducing the flux vector

\[
 f_\varphi = (f,g)^T := v\varphi - D\nabla \varphi, \tag{4.1}
\]

the problem we consider can be represented in integral formulation as: determine \( \varphi \in H^1(\Omega) \) with \( \varphi|_{\partial \Omega} = \psi \), such that

\[
 \oint_{\partial V} f_\varphi \cdot n \, d\gamma = \int_V s \, d\Omega, \quad \text{for all } V. \tag{4.2}
\]

Here we used standard notation for the Sobolev space \( H^1(\Omega) \). In this chapter, we study a finite volume discretization technique based on a combination of finite volume discretizations on several uniform grids with different grid sizes. For the discretization of the convection-diffusion problem on the uniform grids, we consider a standard vertex-centered finite volume discretization method. This choice is merely for notational convenience. In practice, one would most likely follow a cell-centered approach, as this ensures conservation at the domain boundaries too. The technique we present may be generalized, however, to cell-centered methods and to so-called structured boundary conforming grids (cf. [Wesseling, 1991]) or to logically rectangular grids.

The finite volume discretizations on the uniform grids are standard and can be found in many textbooks; the presentation, however, is adapted to the generalization to composite grids. We use a grid size parameter \( H = 1/N \), \( N \) a positive integer, and grid points \( (x_i,y_j) := (iH, jH), (x_{i+1/2},y_j) := ((i+1/2)H, jH), (x_i,y_{j+1/2}) := (iH, (j+1/2)H), \) \( i, j \) positive integers. In a vertex-centered approach one uses a computational grid \( \Omega^H \) defined by

\[
 \hat{\Omega}^H := \{(x_i,y_j)\} \cap \bar{\Omega}, \quad \partial \hat{\Omega}^H := \hat{\Omega}^H \cap \partial \Omega, \quad \Omega^H := \hat{\Omega}^H \setminus \partial \hat{\Omega}^H, \tag{4.3}
\]
4.1 Problem formulation and finite volume discretization on a uniform grid

and a control volume \( V_{ij} \) around each grid point in \( \Omega^H \)

\[
V_{ij} := (x_{i-1/2},x_{i+1/2}) \times (y_{j-1/2},y_{j+1/2}).
\]  

(4.4)

The midpoints of the interface of these volumes form a dual grid

\[
V^H := \left(\{(x_{i+1/2},y_j)\} \cup \{(x_i,y_{j+1/2})\}\right) \cap \Omega,
\]  

(4.5)

on which we will define discrete fluxes. The spaces of grid functions on \( \Omega^H, \tilde{\Omega}^H, V^H \) are denoted by \( G(\Omega^H), G(\tilde{\Omega}^H), G(V^H) \), respectively. For grid functions \( F^H \in G(\Omega^H) \), we write \( F^H = (F_{ij}^H)_{i,j = 1,2,\ldots,N-1} \), with \( F_{ij}^H := F^H(x_i,y_j) \). We use a similar notation for elements in \( G(\tilde{\Omega}^H), G(V^H) \). We introduce, for \( F^H \in G(V^H) \), central difference operators \( \nabla_x^H, \nabla_y^H: G(V^H) \to G(\Omega^H) \) by

\[
(\nabla_x^H F^H)_{i,j} := F^H_{i+1/2,j} - F^H_{i-1/2,j}, \quad (\nabla_y^H F^H)_{i,j} := F^H_{i,j+1/2} - F^H_{i,j-1/2}.
\]  

(4.6)

We define the integrated flux \( F(\varphi) \in G(V^H) \) as follows (cf. (4.1)):

\[
F_{i+1/2,j} := \int_{y_{j-1/2}}^{y_{j+1/2}} f(x_{i+1/2},\eta) \, d\eta, \quad F_{i,j+1/2} := \int_{x_{i-1/2}}^{x_{i+1/2}} g(\xi,y_{j+1/2}) \, d\xi.
\]  

(4.7)

Note that this is the integrated flux over the interfaces of the control volumes \( V_{ij} \) as in (4.4). Finally, we define \( S \in G(\Omega^H) \) by

\[
S_{ij} := \int_{V_{ij}} s \, d\Omega.
\]  

(4.8)

Applying the conservation law in (4.2) with \( V = V_{ij} \) yields:

\[
(\nabla_x^H F(\varphi))_{ij} + (\nabla_y^H F(\varphi))_{ij} = S_{ij}.
\]  

(4.9)

In finite volume discretizations the continuous fluxes in (4.7), which depend on the continuous solution \( \varphi \), are approximated using a finite difference scheme. For \( \xi \in G(\tilde{\Omega}^H) \), we introduce a discrete flux grid function \( F^H(\xi) \in G(V^H) \). Here we use a general setting and we will not be specific about the particular finite difference scheme that is used. We only assume that the difference scheme \( F^H(\xi) \) is local and linear in \( \xi \), i.e.,

\[
F^H_{i+1/2,j}(\xi) = \sum_{k=0,1, m=-1,0,1} \alpha_{i+k,j+m} \xi_{i+k,j+m},
\]  

(4.10)

with given real coefficients \( \alpha_{pq} \). We use a similar approximation \( F^H_{i+1/2,j} \). In practice, the integral in (4.8) is approximated by a quadrature rule. The resulting approximation of \( S \) is denoted by \( S^H \).

Example 4.1 If we apply the midpoint rule to approximate the integrals in (4.7), (4.8), and use central differences to approximate the fluxes at midpoints of volume faces, we obtain for \( F^H(\xi) \) and \( S^H \)

\[
F^H_{i+1/2,j} = f^H_{i+1/2,j} H, \quad F^H_{i,j+1/2} = g^H_{i,j+1/2} H, \quad S^H_{ij} = s(x_i,y_j) H^2,
\]
where
\[ f_{i+1/2,j}^H = v_1(x_{i+1/2}, y_j) \frac{\xi_{ij} + \xi_{i+1,j}}{2} - D(x_{i+1/2}, y_j) \frac{\xi_{i+1,j} - \xi_{ij}}{H} , \]
\[ g_{i,j+1/2}^H = v_2(x_i, y_{j+1/2}) \frac{\xi_{ij} + \xi_{i,j+1}}{2} - D(x_i, y_{j+1/2}) \frac{\xi_{i,j+1} - \xi_{ij}}{H} . \]

In the above, \( \xi \in G(\tilde{\Omega}^H) \), \( F^H(\xi) \in G(V^H) \), and \( S^H \in G(\Omega^H) \).

In (4.9), we replace the continuous fluxes \( F \) by approximate fluxes \( F^H \) as in (4.10) and \( S_{ij} \) by \( S_{ij}^H \). We then obtain a finite volume discretization which can be represented as: find \( \varphi^H \in G(\tilde{\Omega}^H) \) such that
\[
\begin{cases} 
\nabla^H_x F^H(\varphi^H) + \nabla^H_y F^H(\varphi^H) = S^H, \\
\varphi^H = \psi \text{ on } \partial \tilde{\Omega}^H. 
\end{cases}
\]

(4.11)

This discretization yields \((N-1)^2\) equations for the \((N-1)^2\) unknown values of \( \varphi^H \) on \( \Omega^H \).

### 4.2 An iterative finite volume discretization on composite grids

In this section, we will present a finite volume method for approximating the continuous solution \( \varphi \) on a composite grid. In Section 4.2.1, we explain how a composite grid is formed by combining two or more uniform grids with different grid sizes. In Section 4.2.2, we adapt the general LDC method from Hackbusch, 1984 to a finite volume setting. The LDC method is an iterative method, hence we obtain an iterative finite volume discretization method. In Section 4.2.3, we derive some properties of the method. In particular it is shown that a conservation property holds on the composite grid.

#### 4.2.1 Composite grid

In this section we recall the concept of composite grids and introduce some notation. The grids we consider result from a uniform basis grid with grid size \( H \), cf. Section 4.1, that is extended with a region of locally uniform refinement \( \Omega_l \), which is such that it contains the part(s) of \( \Omega \) where relatively high resolution is needed. In Section 4.3, an example of an interface problem is given, where it is a priori clear that in a (small) part of the domain \( \Omega \) a much higher resolution is required than in the remaining part. Further examples can be found in Ferret, 1996, Ferret and Reusken, 1996b, Hackbusch, 1984.

We assume that \( \Omega_l \subset \Omega \) is chosen such that
\[
(x_i, y_j) \in \Omega^H \cap \Omega_l \iff W_{ij} \subset \Omega_l 
\]
(4.12)
4.2 An iterative finite volume discretization on composite grids

![Composite grid illustration](image)

Figure 4.1: A composite, global coarse and local fine grid; $H = 1/6$, $N = 5$, and the refinement factor $\sigma$ equals 3. Grid points, control volumes, and fluxes are denoted by little circles, large squares, and arrows, respectively. The shaded region is $\Omega_l$; the points of $\Gamma^H$ are marked by filled circles.

holds with

$$W_{ij} := (x_{i-1}, x_{i+1}) \times (y_{j-1}, y_{j+1}).$$

(4.13)

Note that $V_{ij} \subset W_{ij}$, so that $W_{ij} \subset \Omega_l$ implies $V_{ij} \subset \Omega_l$. Also, $\Omega_l$ is not a union of control volumes $V_{ij}$. In $\Omega_l$ we apply, as in $\Omega$, a vertex-centered finite volume method, i.e., we first introduce a uniform computational grid with grid size $h < H$. As before, the local fine grid is denoted by $\Omega^h_l$. To make sure that grid points in $\Omega^H_l := \Omega^H \cap \Omega_l$ are grid points of $\Omega^h_l$, and that boundaries of control volumes in the local fine grid coincide with boundaries of control volumes in the global coarse grid, we assume the refinement factor

$$\sigma := \frac{H}{h}$$

(4.14)

to be an odd integer. We emphasize that a refinement factor $\sigma \gg 1$ is allowed, i.e., we can use a global coarse grid and a local fine grid with different resolution properties. In Figure 4.1 an example of a composite grid is shown (cf. also Section 4.3). As before, the interface between the global coarse grid and the local fine grid will be denoted by $\Gamma := \partial \Omega_l \setminus \partial \Omega$, and we will use the sets $\Gamma^H := \Gamma \cap \Omega^H$, $\Gamma^{H,h} := \Omega^H \cup \Omega^h$, and $\bar{\Omega}^{H,h} := \bar{\Omega}^H \cup \bar{\Omega}^h$. Note that the boundaries of the physical domain do not coincide with the edges of control volumes. This is due to the fact that we have used a vertex-centered method on the uniform grids. We recall our remark about this approach from the previous section: it is merely for notational convenience and the technique may easily be generalized to cell-centered methods. Our interest here is conservation of the finite volume discretization across the interface $\Gamma$.

4.2.2 A local defect correction finite volume discretization

Basically, the LDC iteration in [Hackbusch, 1984] consists of the following steps:

1. Solve a global coarse grid problem with given right hand side.
2. Solve a local fine grid problem with artificial boundary conditions on the interface \( \Gamma \).

3. Compute a defect correction term for the right hand side of the coarse grid problem, and go to 1.

Below, we discuss these three steps, resulting in Algorithm 4.2. For adapting the general LDC method to a finite volume setting, the key point is a defect correction (Step 3) which is based on discretization error estimates for the numerical fluxes.

We first compute an initial approximation \( \varphi^H \) on the global coarse grid using the standard finite volume discretization of Section 4.3, i.e., \( \varphi^H \) is the solution of the discrete problem (4.11). Next, we formulate a boundary value problem on the local domain \( \Omega_l \) using the coarse grid approximation to define artificial Dirichlet boundary conditions on the interface \( \Gamma \). To determine the artificial boundary conditions, we use an interpolation operator \( P_h^H \) as before. At \( \partial \Omega_l \cap \partial \Omega \) we use the given Dirichlet boundary conditions. We are thus led to an analogon of problem (4.2) on the subdomain \( \Omega_l \) with artificial boundary conditions on the interface \( \Gamma \). We discretize this problem on the uniform local grid \( \Omega^h_l \) using the method described in Section 4.1. We use a notation in which local fine grid quantities are denoted by a subscript \( h \) and a superscript \( h \), e.g.: \( \Omega^h_l \) (computational grid on \( \Omega_l \)), \( V^h_l \) (dual grid on \( \Omega_l \)), \( G(V^h_l) \) (grid functions on \( V^h_l \)) and, for \( \xi \in G(V^h_l) \), \( F^h_l(\xi) \in G(V^h_l) \) (discrete flux on \( V^h_l \)). These quantities related to \( \Omega_l \) are defined in exactly the same way as their analogons in Section 4.1 which are related to \( \Omega \). Using this notation the discrete local fine grid problem can be formulated as: find \( \varphi^l_h \in G(V^h_l) \) such that

\[
\left\{ \begin{array}{l}
\nabla^h_l F^h_l(\varphi^l_h) + \nabla^h_y F^h_y(\varphi^h) = S^l_h, \\
\varphi^l_h = \psi \text{ on } \partial \Omega^h_l \cap \partial \Omega, \\
\varphi^h_l = p^h,h(\varphi^H|_{\Omega_l}) \text{ on } \partial \Omega^h_l \cap \Gamma.
\end{array} \right.
\]

The discrete solutions \( \varphi^H \) and \( \varphi^h \) yield an approximation of \( \varphi \) at all points of the composite grid. We denote this by \( \varphi^{H,h} \) and take the newest values in grid points belonging to both the coarse grid and the fine grid, viz.

\[
\varphi^{H,h} := \left\{ \begin{array}{ll}
\varphi^h_l, & \text{on } \Omega^h_l, \\
\varphi^H, & \text{on } \Omega^H \setminus \Omega^h_l.
\end{array} \right.
\]

We now derive the third step in the algorithm, in which we use the (more accurate) approximation found on the local fine grid to compute a correction for the right hand side of the global coarse grid problem. Substitution of the continuous solution \( \varphi \) into (4.11) yields a defect

\[
d^H := \nabla^H_x F^H(\varphi|_{\Omega^H}) + \nabla^H_y F^H(\varphi|_{\Omega^H}) - S^H.
\]

Combination of (4.9) and (4.17) yields:

\[
d^H = \nabla^H_x F^H(\varphi|_{\Omega^H}) - F(\varphi) + \nabla^H_y F^H(\varphi|_{\Omega^H}) - F(\varphi) - (S^H - S).
\]

This expression for the coarse grid defect is used to derive an approximation for \( d^H \) by estimating the flux discretization error \( F^H(\varphi|_{\Omega^H}) - F(\varphi) \) and the source term discretization error \( S^H - S \).
After solving the global coarse and local fine grid problems, the following approximations are available for an arbitrary horizontal flux $F_{i+1/2,j}(\phi)$:

1. the coarse grid approximation of the flux, $F^H_{i+1/2,j}(\phi)$;
2. a coarse grid approximation of the flux based on the most recently calculated approximation for $\phi$, i.e., $F^H_{i+1/2,j}(\phi^H,h|_{\Omega^H})$;
3. a sum of fine grid approximations of fluxes,

$$F_{i+1/2,j}^{\text{sum}}(\phi^h) := \frac{(\sigma-1)/2}{2} \sum_{k=-(\sigma-1)/2}^{(\sigma-1)/2} F_{i+1/2,j+k}(\phi^h). \quad (4.19)$$

This third approximation exists, if $(x_{i+1/2},y_j) \in \Omega_l$, which means that the cell face $x = x_{i+1/2}$, $y_j-1/2 \leq y \leq y_j+1/2$ lies in the area of refinement.

Note that both in the second and third approximation, we use the solution $\phi^h$ of the discrete local fine grid problem (4.15). In the second approximation, however, only a coarse grid flux discretization $F^H$ is used, whereas in the third approximation, a fine grid flux discretization $F^h$ is used. These three approximations are considered to be listed in order of increasing accuracy. Because similar approximations are available for the other fluxes, $F_{i-1/2,j}(\phi)$, $F_{i,j+1/2}(\phi)$, $F_{i,j-1/2}(\phi)$, we can define a coarse grid flux vector which uses information from the local fine grid solution: $F^{\text{best}}(\phi^H,h) \in G(\Omega^H)$ is defined by:

$$F^{\text{best}}(\phi^H,h) := \begin{cases} F_{i+1/2,j}^{\text{sum}}(\phi^h), & \text{on } \Omega^H \cap \Omega_l \text{ (as in (4.19)),} \\ F^H(\phi^H,h|_{\Omega^H}), & \text{elsewhere.} \end{cases} \quad (4.20)$$

We use this flux vector to give the following flux discretization error estimate

$$F^H(\phi|_{\Omega^H}) - F(\phi) \approx F^H(\phi^H,h|_{\Omega^H}) - F^{\text{best}}(\phi^H,h) =: d^H_F(\phi^H,h). \quad (4.21)$$

Analogously, we have the following approximations for an arbitrary source term $S_{ij}$:

1. the coarse grid approximation of the source term, $S^H_{ij}$;
2. a sum of fine grid approximations of source terms,

$$S_{i,j}^{\text{sum}} := \sum_{k=-(\sigma-1)/2}^{(\sigma-1)/2} \sum_{m=-(\sigma-1)/2}^{(\sigma-1)/2} S^h_{i,kh,j+mh}. \quad (4.22)$$

This second approximation only exists, if $(x_i,y_j) \in \Omega_l$, i.e., if the control volume $V_{ij}$ lies in the area of refinement.

Again, the last approximation is considered to be most accurate, and we define $S^{\text{best}} \in G(\Omega^H)$ by

$$S^{\text{best}} := \begin{cases} S_{i,j}^{\text{sum}}, & \text{on } \Omega^H \text{ (as in (4.22)),} \\ S^H, & \text{elsewhere.} \end{cases} \quad (4.23)$$
We use this source term vector to give the following source term discretization error estimate

\[ S^H - S \approx S^H - S_{\text{best}} \approx d_S^H. \quad (4.24) \]

Using (4.21) and (4.24) to estimate \( d_H^H \) in (4.18), we propose

\[ d_H^H(\varphi_H^H, h) := \nabla^H x d^H_F(\varphi_H^H) + \nabla^H y d^H_F(\varphi_H^H) - d_S^H \quad (4.25) \]

as a defect correction term in the right hand side of the coarse grid problem. Hence, we introduce the following notation for \( \varphi_H^H, h \in G(\bar{\Omega}_H^h) \):

\[ S_H^H(\varphi_H^H, h) := S^H + d_H^H(\varphi_H^H, h). \quad (4.26) \]

Using the updated right hand side (4.26), we can repeat the procedure described above, i.e., solve a coarse grid problem, define artificial boundary conditions on \( \Gamma \), solve a local fine grid problem, etc. This results in the following iterative method.

**Algorithm 4.2**

**LDC algorithm with finite volume adapted correction term**

**Initialization**

- Solve the basic coarse grid problem (4.11) for \( \varphi_H^0 \in G(\bar{\Omega}_H^0) \).
- Solve the local fine grid problem (4.15) for \( \varphi_{l,0}^h \in G(\bar{\Omega}_l^h) \).
- Define the composite grid approximation \( \varphi_{l-1,0}^H \in G(\bar{\Omega}_H^l) \) as in (4.16).

**Iteration, \( k = 1, 2, \ldots \)**

- Compute an updated right hand side \( S_H^H(\varphi_{l-1,0}^H) \) as in (4.26).
- Solve the global coarse grid problem: find \( \varphi_H^k \in G(\bar{\Omega}_H^k) \) such that

\[ \begin{aligned}
\nabla^H x F_H^H(\varphi_H^k) + \nabla^H y F_H^H(\varphi_H^k) &= S_H^H(\varphi_{l-1,0}^H), \\
\varphi_H^k &= \psi \text{ on } \partial \bar{\Omega}_H^k.
\end{aligned} \quad (4.27) \]

- Solve the local fine grid problem: find \( \varphi_{l,k}^h \in G(\bar{\Omega}_l^h) \) such that

\[ \begin{aligned}
\nabla^h x F_l^h(\varphi_{l,k}^h) + \nabla^h y F_l^h(\varphi_{l,k}^h) &= S_l^h, \\
\varphi_{l,k}^h &= \psi \text{ on } \partial \bar{\Omega}_l^h \cap \partial \Omega, \\
\varphi_{l,k}^h &= P_h(\varphi_k^H|_{\Gamma_l^h}) \text{ on } \partial \bar{\Omega}_l^h \cap \Gamma.
\end{aligned} \quad (4.28) \]

- Define the composite grid approximation

\[ \varphi_k^{H,h} := \begin{cases} 
\varphi_{l,k}^h, & \text{on } \Omega_l^h, \\
\varphi_k^H, & \text{on } \Omega^H \setminus \Omega_l^H.
\end{cases} \quad (4.29) \]

This is the LDC method as presented in Chapter 3, but now adapted to a setting with finite volume discretizations. In particular, the form of the updated right hand side \( S_H^H(\varphi_{k-1}^H) \) is new. Here, the correction term is chosen such that in the limit (\( k \to \infty \))
the resulting composite grid discretization is still conservative; this is discussed in Section 4.2.3. In Section 4.3.2, we will show that for certain problems the standard choice for the correction term (as used in [Ferket and Reusken, 1996a; Ferket and Reusken, 1996b; Hackbusch, 1984]) may yield a poor discretization, because the discretization is not conservative on the composite grid, whereas with the finite volume adapted correction term presented here the results are satisfactory.

For grid points of the coarse grid that are not located at the interface, the computation of $S^H(p^{H,H}_k)$ can be simplified using the results in Lemma 4.3. For coarse grid points $(x_i, y_i)$ located within the area of refinement, i.e., the control volume $V_{ij} \subset \Omega_t$, the simplification is achieved by summation of the conservation laws that hold on the fine grid control volumes that form $V_i$. For coarse grid points $(x_i, y_i)$ with control volumes outside the area of refinement, i.e., $V_i \subset \Omega \setminus \Omega_t$, the updated right hand side equals the original right hand side.

**Lemma 4.3** For $S^H(p^{H,H}_k)$ as used in (4.27), we have, with $\Omega^H_t = \Omega^H \setminus (\Omega^H \cup \Gamma^H)$:

$$S^H_i(\varphi^H_k) = \begin{cases} (\nabla^H_x \cdot \mathbf{f}^H(\varphi^H_k) + \nabla^H_y \cdot \mathbf{f}^H(\varphi^H_k)\mid_{\Omega^H})_{ij}, & \text{for } (x_i, y_j) \in \Omega^H, \\ S^H_i(\varphi^H_k), & \text{for } (x_i, y_j) \in \Omega^H \setminus \Omega^H_t \end{cases}$$

(4.30)

**Proof.** Consider a grid point $(x_i, y_j) \in \Omega^H_t$. Adding the fine grid equations in (4.28) for all fine grid points in the coarse grid control volume $V_{ij}$ and replacing $k$ by $k-1$, we find the following conservation property over this control volume:

$$(\nabla^H_x \cdot \mathbf{f}^{\text{sum}}(\varphi_{i,k-1}) + \nabla^H_y \cdot \mathbf{f}^{\text{sum}}(\varphi_{i,k-1}))_{ij} = S^H_i(\varphi_{i,k-1}).$$

Using the notation in (4.20), (4.21), and (4.24) we now have for $(x_i, y_j) \in \Omega^H_t$:

$$(S^H(p^{H,H}_{k-1}))_{ij} = S^H_i + (\nabla^H_x \cdot \mathbf{d}^H(\varphi^H_{k-1}) + \nabla^H_y \cdot \mathbf{d}^H(\varphi^H_{k-1}))_{ij} - (S^H_i - S^H_i^{\text{sum}}(x_i, y_j))$$

$$= (\nabla^H_x \cdot \mathbf{f}^H(\varphi^H_{k-1}) + \nabla^H_y \cdot \mathbf{f}^H(\varphi^H_{k-1})\mid_{\Omega^H})_{ij} + S^H_i^{\text{sum}}(x_i, y_j) - (\nabla^H_x \cdot \mathbf{f}^H(\varphi^H_{k-1}) + \nabla^H_y \cdot \mathbf{f}^H(\varphi^H_{k-1}))_{ij}$$

$$= (\nabla^H_x \cdot \mathbf{f}^H(\varphi^H_{k-1}) + \nabla^H_y \cdot \mathbf{f}^H(\varphi^H_{k-1})\mid_{\Omega^H})_{ij},$$

which proves the first part of (4.30). From the definitions in (4.20) and (4.21), we obtain that $\mathbf{d}^H(\varphi^H_{k-1})$ equals zero on $\nabla^H \cap (\Omega \setminus \Omega_t)$, and hence the difference operators $\nabla^H_x$ and $\nabla^H_y$ applied to this grid function yield zero on $\Omega^H_t$. This gives the second part of (4.30).

Note that in (4.30) we have formulas for $S^H(p^{H,H}_k)$ for $(x_i, y_j) \in \Omega^H \cup \Omega^H_t = \Omega^H \cup \Gamma^H$ in which the sum of fine grid fluxes $F^{\text{sum}}_i(\varphi^H_{k-1})$ is not needed. Such sums of fine grid fluxes have to be computed on faces of control volumes $V_{ij}$ with $(x_i, y_j) \in \Gamma^H$ only. Also note that the term $S^H_i^{\text{sum}}$ can be avoided in the computation of $S^H(p^{H,H}_{k-1})$.

The method presented in this section has a straightforward generalization to logically rectangular grids. Also, for the method to be applicable to three-dimensional problems, only minor modifications are needed.
4.2.3 Properties of the LDC method

The LDC algorithm that is described in Section 4.2.2 is an iterative process, which implicitly gives a discretization of the convection-diffusion problem on a composite grid. In this section, we discuss a few properties of this discretization. Throughout this section, we will assume that the LDC iteration converges. Numerical experiments (cf. Section 4.3 and theoretical results in Chapter 5 and in [Ferket, 1996, Ferket and Reusken, 1996a, Hackbusch, 1984]) support this assumption. A sufficient condition for the iterative process to be convergent is

\[ \phi^H_k \to \phi^H \quad (k \to \infty), \]

because this implies \( \phi^H|_{\Gamma^H} \to \phi^H|_{\Gamma^H} \quad (k \to \infty) \), and therefore \( \phi^H_{i,k} \to \phi^H_{i,*} \quad (k \to \infty) \). From these two limit solutions \( \phi^H \in G(\Omega^H) \) and \( \phi^H_{i,*} \in G(\Omega^H) \), we define a composite grid approximation \( \phi^{H,h} \in G(\Omega^{H,h}) \) as in (4.29). In Lemma 4.4 below, it is shown that the coarse grid solution \( \phi^H \) and the local fine grid solution \( \phi^H_{i,*} \) coincide in \( \Omega^H \).

**Lemma 4.4** Assume that the local coarse grid homogeneous system

\[
\begin{aligned}
\{ & \nabla_x^H F^H(v) + \nabla_y^H F^H(v) = 0 \text{ on } \Omega^H, \\
& v = 0 \text{ on } \partial \Omega^H, \\
\end{aligned}
\]

has only the zero solution in \( G(\Omega^H) \). Then the limit solution \((\phi^H, \phi^H_{i,*})\) of the LDC iteration satisfies

\[ \phi^H|_{\Omega^H} = \phi^H_{i,*}|_{\Omega^H}. \]

**Proof.** From (4.27) and (4.30), we obtain, for \((x_i, y_j) \in \Omega^H,\)

\[
\left( \nabla_x^H F^H(\phi^H) + \nabla_y^H F^H(\phi^H) \right)_{ij} = \left( \nabla_x^H F^H(\phi^H|_{\Omega^H}) + \nabla_y^H F^H(\phi^H_{i,*}|_{\Omega^H}) \right)_{ij}.
\]

Note that \( \phi^{H,h}(x_i, y_j) = \phi^H(x_i, y_j) \) for \((x_i, y_j) \in \Gamma^H\) and \( \phi^{H,h}(x_i, y_j) = \phi^H_{i,*}(x_i, y_j) \) for \((x_i, y_j) \in \Omega^H\). Hence, \( v := \phi^H - \phi^{H,h}|_{\Omega^H} \in G(\Omega^H) \) satisfies the system (4.32). From the assumption it follows that this system only has the zero solution, hence \( v = 0 \) on \( \Omega^H \), which is equivalent to (4.33).

We will now discuss the conservation property which holds for the limit solution \( \phi^{H,h} \) on the composite grid. Summation of the conservation laws for individual control volumes \( V_{ij} \), cf. (4.29), leads to a conservation law on the union of these control volumes. This holds, because fluxes over internal faces cancel. Consider, e.g., control volumes \( V_{ij}, V_{i+1,j} \) with \((x_i, y_j) \in \Omega^H, \ (x_{i+1}, y_j) \in \Omega^H\). We have

\[
\int_{\partial V_{ij}} \mathbf{f} \cdot \mathbf{n} \, d\gamma + \int_{\partial V_{i+1,j}} \mathbf{f} \cdot \mathbf{n} \, d\gamma = \int_{\partial(V_{ij} \cup V_{i+1,j})} \mathbf{f} \cdot \mathbf{n} \, d\gamma,
\]

which implies, that summation of the conservation laws in \( V_{ij} \) and \( V_{i+1,j} \) leads to (4.2) with \( V = V_{ij} \cup V_{i+1,j} \). The finite volume discretization on a uniform grid as described in Section 4.1 satisfies the discrete equivalent of (4.34) as is easily seen by adding the discrete conservation laws in (4.11). Therefore, discrete conservation holds for any sub-domain of \( \Omega \) which can be covered with control volumes \( V_{ij} \). A similar result holds for the limit solution \( \phi^{H,h} \) on the composite grid, as is shown in the following theorem.
4.2 An iterative finite volume discretization on composite grids

Figure 4.2: The discretization on the composite grid given by the LDC algorithm with the standard choice for the correction term (left figure) and with the finite volume adapted correction term (right figure).

Theorem 4.5 Under the assumption of Lemma 4.4, the limit solution $\varphi_{H,h}^* \in G(\bar{\Omega}^{H,h})$ satisfies the following system of discrete conservation laws:

$$\nabla_x^H F_{\text{best}}(\varphi_{H,h}^*) + \nabla_y^H F_{\text{best}}(\varphi_{H,h}^*) = S_{\text{best}},$$

with $F_{\text{best}}(\varphi_{H,h}^*)$ defined as in (4.20) and $S_{\text{best}}$ defined as in (4.23).

Proof. Using (4.26) and (4.27), we find

$$\nabla_x^H f_H^\text{H}(\varphi_{H,h}^*) + \nabla_y^H f_H^\text{H}(\varphi_{H,h}^*) = S^H + \nabla_x^H d_H^\text{H}(\varphi_{H,h}^*) + \nabla_y^H d_H^\text{H}(\varphi_{H,h}^*) - d_{S}^H. \quad (4.36)$$

For $d_H^\text{H}(\varphi_{H,h}^*)$, we have, using (4.21) and Lemma 4.4

$$d_H^\text{H}(\varphi_{H,h}^*) = f_H^\text{H}(\varphi_{H,h}^*|_{\Omega'}) - f_{\text{best}}(\varphi_{H,h}^*) = f_H^\text{H}(\varphi_h^*) - f_{\text{best}}(\varphi_{H,h}^*). \quad (4.37)$$

Substitution of (4.37) into (4.36) yields

$$\nabla_x^H F_{\text{best}}(\varphi_{H,h}^*) + \nabla_y^H F_{\text{best}}(\varphi_{H,h}^*) = S^H - d_{S}^H = S_{\text{best}},$$

which proves the theorem. \square

Using Theorem 4.5, it is easily verified, that the discretization given by the LDC algorithm as described in Section 4.2.2 satisfies a discrete equivalent of (4.34), too. Therefore, discrete conservation holds for any subdomain of $\Omega$ which can be covered with control volumes $V_{ij}$.

For $(x_i, y_j) \in \Omega_{ij}^H$, the conservation laws in (4.35) reduce to

$$\left(\nabla_x^H f_{\text{sum}}(\varphi_{i,j}^h) + \nabla_y^H f_{\text{sum}}(\varphi_{i,j}^h)\right)_{ij} = S_{\text{sum}}(x_i, y_j). \quad (4.38)$$
This is the same conservation property one would obtain by adding the conservation laws that hold on the \( \sigma^2 \) fine grid control volumes that partition \( V_{ij} \), cf. (4.28). For \((x_i, y_j) \in \Omega^H \setminus (\Omega^I \cup \Gamma^H)\), the components of (4.35) reduce to

\[
(\nabla^H_x F^H(\phi^H) + \nabla^H_y F^H(\phi^H))_{ij} = S^H(x_i, y_j),
\]

which corresponds to the conservation laws of the finite volume discretization on the global coarse grid, cf. (4.11). For \((x_i, y_j) \in \Gamma^H\), the limit discretization of the finite volume adapted LDC algorithm is such, that the discrete influx into \( V_{ij} \) out of a control volume \( V_{km} \), \((x_k, y_m) \in \Omega^H_l\), matches the total discrete outflux from \( V_{km} \) into \( V_{ij} \). This is illustrated in the right part of Figure 4.2. If we would use the standard choice for the correction term in the LDC algorithm, the limit discretization would be the same on all control volumes \( V_{ij} \) with \((x_i, y_j) \in \Omega^H \setminus \Omega^H_c = \Omega^H \setminus \Gamma^H\). The discretization would be different, however, on control volumes \( V_{ij} \) with \((x_i, y_j) \in \Gamma^H\); these volumes would be treated in the same way as volumes \( V_{ij} \) with \((x_i, y_j) \in \Omega^I\). The difference between the discretizations given by the two LDC algorithms is clarified in Figure 4.2; its consequences are demonstrated by a numerical experiment in Section 4.3.2.

The system of discrete conservation laws in Theorem 4.5 holds for the fully converged composite grid solution \( \psi^{H,h}_* \). However, in practice, often one or two LDC iterations will suffice to obtain a satisfactory approximation of \( \psi \) on \( \Omega^{H,h} \) due to the high rate of convergence of the method. Typically, one has an error reduction by a factor 10 up to 1,000 in each iteration step (cf. the results in Section 4.3 and in \cite{Feret and Reusken, 1996a, Hackbusch, 1984}).

In Section 4.2.2, a vertex-centered finite volume method has been used for both the discretization on the global coarse grid and on the local fine grid. If we would use a cell-centered method on the global coarse grid, we can cover all of \( \Omega \) with control volumes, which yields global discrete conservation. This approach has been followed in the examples of Section 4.3. Note that, as a consequence, boundary conditions for the local fine grid problem have to be treated as in a vertex-centered method (the artificial boundary conditions) or as in a cell-centered method (the physical boundary conditions). This is illustrated in the left part of Figure 4.3. It is also possible to apply a cell-centered finite volume method in \( \Omega_l \) by choosing the refinement factor \( \sigma = H/h \) to be an even integer. See the right part of Figure 4.3. As before, a refined coarse grid control volume is the union of fine grid control volumes, so that we can define a source term discretization error estimate in a straightforward way. However, the points in \( \Omega^I_l := \Omega^H \cap \Omega_t \) are no longer grid points of \( \Omega^H_l \), so that we need to introduce a restriction \( R : G(\Omega^H_l) \rightarrow G(\Omega^I_l) \) to define flux term discretization error estimates.

### 4.3 Numerical experiments

In this section, we consider two simple numerical experiments: an interface problem in the unit square and a one-dimensional convection-diffusion problem. It should be noted, that the technique presented in Section 4.2 is capable of computing phenomena...
4.3 Numerical experiments

4.3.1 A two-dimensional interface problem

We consider a two-dimensional interface problem. In this experiment, usage of the finite volume adapted correction term is not essential. If we would use the standard correction term as in [Hackbusch, 1984] instead of the new correction term $S_H(\phi^{H,h})$ as in (4.26), we obtain similar results. The experiment shows however that the high convergence rate of the LDC algorithm still holds for its generalization to finite volume discretizations.

We choose the mass flux $v$ equal to zero, and take a diffusion coefficient $D$ that is discontinuous across a curve in $\Omega = (0,1) \times (0,1)$ and has a small value in part of the domain, viz.

$$ D(x,y) := \begin{cases} \delta, & \text{for } (x,y) \in U_\varepsilon, \\ 1, & \text{for } (x,y) \in \Omega \setminus U_\varepsilon, \end{cases} \quad (4.40) $$

where $0 < \delta \ll 1$, $U_\varepsilon := (1/2 - \varepsilon,1/2 + \varepsilon) \times (1/2 - \varepsilon,1/2 + \varepsilon)$, $0 < \varepsilon < 1/2$. The boundary conditions $\psi$ and right hand side $s$ are chosen such that the solution to the continuous problem is known and has a high activity region in $U_\varepsilon$. If we define the auxiliary function $a$ by

$$ a(x) := \exp\left(-(x-1/2)^2\right) - \exp(-\varepsilon^2), \quad (4.41) $$

and set

$$ \psi(x,y) := a(x) a(y), \quad (x,y) \in \partial\Omega, $$
$$ s(x,y) := -a''(x) a(y) - a(x) a''(y), \quad (x,y) \in \Omega, $$

we have the following expression for the solution $\phi$ of the continuous problem:

$$ \phi(x,y) = \begin{cases} \delta^{-1} a(x) a(y), & \text{for } (x,y) \in U_\varepsilon, \\ a(x) a(y), & \text{for } (x,y) \in \Omega \setminus U_\varepsilon. \end{cases} $$
Note that ψ and s depend on ε but not on δ. In the experiment below, we take δ = 10^{-8}, ε = 2/81. The source term s and the exact solution ϕ with these values of δ and ε are shown in Figure 4.4.

Because of the different resolutions needed to represent ϕ in U_ε and Ω \ U_ε, we will use the LDC method with the finite volume adapted correction term as described in Section 4.2.2 to discretize the boundary value problem for ϕ on a composite grid. We take Ω_1 := (1/2 − 1/27, 1/2 + 1/27) × (1/2 − 1/27, 1/2 + 1/27). For the global coarse grid, grid sizes H = 1/3^3, H = 1/3^4, and H = 1/3^5 have been used; the refinement factor σ has been chosen equal to σ = 3, σ = 3^2, and σ = 3^3. In this model problem, the location of the physical interface (∂U_ε) is such that for H = 1/3^k, k ≥ 3, and σ = 3^m, m ≥ 1, this interface is on grid lines in Ω^H. Therefore, a simple central difference flux approximation scheme, as in Example 4.1, can be used (see Wesseling, 1992 for a more detailed discussion of this topic). In a setting where this favorable interface-grid alignment does not hold, other, more advanced, finite volume discretization schemes should be used. The LDC method, however, remains the same.

Because we would like to study the performance of the LDC (outer) iteration, the linear systems arising in the LDC algorithm have been solved to high accuracy using CG iteration with incomplete Cholesky factorization as a preconditioner. The properties shown below still hold, however, if we use low, but “reasonable”, accuracy in the inner iterations.

The numerical results of the LDC method are presented in Table 4.1. Listed are the number of unknowns in the computation and the discretization error in the scaled Euclidean norm

\[ \left\| \phi|_{\Omega^H} - \phi^*_H \right\|_2 / (N - 1), \]

(4.42)

where \((N - 1)^2\) is the number of grid points in Ω^H. Note that diagonal elements in the table correspond to uniform grids. From Table 4.1, we conclude that the LDC algorithm can reduce the discretization error on the global coarse grid (grid size H) to an error that
4.3 Numerical experiments

Table 4.1: Numerical results for the two-dimensional interface problem computed using the LDC algorithm with finite volume adapted correction term on a composite grid. The global coarse grid has grid size \( H \); the local fine grid has grid size \( h \).

<table>
<thead>
<tr>
<th>( H ) = ( 1/3^3 )</th>
<th>( H ) = ( 1/3^4 )</th>
<th>( H ) = ( 1/3^5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unknowns</td>
<td>Error</td>
<td>Unknowns</td>
</tr>
<tr>
<td>( h = 1/3^4 )</td>
<td>729</td>
<td>( 1.7 \cdot 10^{-9} )</td>
</tr>
<tr>
<td>( h = 1/3^5 )</td>
<td>1,090</td>
<td>( 1.6 \cdot 10^{-5} )</td>
</tr>
<tr>
<td>( h = 1/3^6 )</td>
<td>3,754</td>
<td>( 1.2 \cdot 10^{-5} )</td>
</tr>
</tbody>
</table>

is of the same order of magnitude as the error on a global uniform grid with grid size \( h \), using considerably less grid points than a computation on a global uniform grid with grid size \( h \) would require. This is, e.g., illustrated by the computation on the composite grid with grid sizes \( H = 1/3^4 \), \( h = 1/3^5 \), which uses only 6,922 grid points to find an approximation with the same error as a computation on a uniform grid with grid size \( H = 1/3^5 \), which involves 59,049 unknowns. Note that even the error in the result on the composite grid with \( H = 1/3^3 \), \( h = 1/3^5 \), which has only 1,090 grid points, is already of the same order of magnitude.

Finally, we remark that the uniform grid with grid size \( H = 1/3^3 \) completely misses the high activity region \( U_\epsilon \), causing a very large discretization error. This error is reduced by a factor of order \( 10^4 \) by refining the high activity zone with a factor \( \sigma \) of only 3 (introducing just 49 new grid points). The excellent rate of convergence of the LDC method is illustrated by the fact that the results in Table 4.1 are already found after just one LDC correction step. In other words, a table listing the discretization error \( \| \phi_{\Omega_H} - \phi_H \|_2 / (N - 1) \), would be the same as Table 4.1.

We stated in the introduction of this section that we would obtain similar results if we would use the standard correction term as in \cite{Hackbusch, 1984} instead of the new correction term \( S_H(\phi_{H,h}) \) as in (4.26). This is not surprising, since the conservation property is crucial across the physical interface \( \partial U_\epsilon \), but of minor importance across the artificial interface \( \Gamma \). Hence, using a finite volume discretization for the local fine grid problem is of major importance, but using the new correction term, which yields conservation across \( \Gamma \), is of minor importance.

4.3.2 A one-dimensional convection-diffusion problem

In this section, we treat a very simple one-dimensional problem, in which global conservation is crucial. For this problem, the results of the classical LDC algorithm as in \cite{Feret and Reusken, 1996a, Feret and Reusken, 1996b, Hackbusch, 1984} and in Chapter 3 are very poor, whereas the finite volume adapted algorithm yields satisfactory results.

We consider a time-dependent convection-diffusion problem, which is a model for the behavior of water held inside a basin by two levies. We choose the following values for the parameters in the problem. The diffusion coefficient \( D \) equals 1 in \( \Omega = (0,1) \).
An LDC method for finite volume discretizations

The mass flux is time-dependent: 
\[ v(x, t) := 5 + 10 \sin(8\pi t), \quad x \in \Omega, \quad t \geq 0. \]
There is no production or consumption in the domain, hence 
\[ s(x) := 0, \quad x \in \Omega. \]
This leads to the following partial differential equation for \( \phi \) in \( \Omega \):
\[
\frac{\partial \phi}{\partial t} + \frac{\partial}{\partial x} (f \phi) = 0, \quad f \phi := v \phi - \frac{\partial \phi}{\partial x}, \quad (4.43)
\]
which expresses the tendency of the water level \( \phi \) to follow the wind \( v \), and to level out. We choose flux boundary conditions, viz. 
\[ f \phi(0, t) = 0, \quad f \phi(1, t) = 0, \]
which model the two levies that prevent the water from flowing in or out of the basin. The initial condition is 
\[ \phi(x, 0) = 1. \]
Integration of the partial differential equation over \( \Omega \) yields the global conservation law
\[
\frac{d}{dt} \int_{\Omega} \phi(x, t) \, dx = 0. \quad (4.44)
\]
We first applied the Euler Backward method for the time discretization. In each Euler step, a continuous two-point boundary value problem has to be solved. Because of boundary layer effects, the water level varies most in the outer parts of the spatial domain, i.e., in \( (0, \delta) \) and in \( (1 - \delta, 1) \). For this reason we will use a composite grid for space discretization. The composite grid used consists of a global coarse grid with grid size \( H = 1/20 \) in \( \Omega \) and two local fine grids, both with grid size \( h = 1/100 \), in \( \Omega_{l,1} := (0, 1/8) \) and in \( \Omega_{l,2} := (7/8, 1) \). We present results for both the LDC method with the standard choice for the correction term and the LDC method with the finite volume adapted correction term. The results are shown in Figure 4.5.

Clearly, the LDC method with the standard choice for the correction term leads to an unrealistic and decreasing water level through “numerical evaporation”. The LDC method with the finite volume adapted correction term satisfies a discrete equivalent of the global conservation law \( (4.44) \), and preserves the water level, cf. Figure 4.5(c).
Chapter 5

Convergence behavior of the LDC method as an iterative process

This chapter focuses on the behavior of the LDC algorithm as an iterative method. In his original paper [Hackbusch, 1984], Hackbusch sketches a general strategy for a convergence proof based on interior regularity of the discrete solution on the local fine grid. Ferket [Ferket, 1996] derives the system of equations that the fixed point of the LDC iteration satisfies, if it exists. He also gives an expression for the iteration matrix of the LDC iteration, but he does not give bounds for the spectral radius or norm of this matrix. In this chapter, we present a new expression for the iteration matrix \( M \) of the LDC iteration. We derive for a model problem (Poisson’s equation on the unit square with one area of refinement) the following upper bound for the norm of the iteration matrix:

\[
\| M \|_\infty \leq C H^2,
\]

in which \( C \) is a constant and \( H \) is the grid size of the global coarse grid.

Convergence of the LDC iteration in a finite element context was studied by Wappler [Wappler, 1999]. He gives a convergence proof of the LDC iteration in a variational setting. To accomplish his proof, Wappler has to make some restrictive assumptions on the interpolation and restriction operators in the LDC algorithm. Just like in the finite difference context, he finds the rate of convergence of the iteration to improve when the grid size of the coarse grid tends to zero.

In this chapter, we consider the LDC algorithm as formulated in Chapter 3. Using some properties that have been proved in Section 3.3, we derive an expression for the iteration matrix. In Section 5.2, we consider the algorithm for a model problem, Poisson’s equation on the unit square. For the coarse grid discretization, we use the standard
five-point stencil for the Laplacian; on the local region, we solve a continuous boundary value problem. Trigonometric interpolation is used on the interface between the coarse grid and the local region. It is shown that the norm of the iteration matrix $M$ satisfies the upper bound (5.1). In Section 5.3, we consider the same model problem as in Section 5.2, but here we use the standard five-point stencil for the Laplacian on a uniform fine grid in the local region. We derive an upper bound for the norm of the iteration matrix as in (5.1). In Section 5.4, we verify the theoretical results of Sections 5.2 and 5.3 in some numerical experiments.

5.1 Expression for the iteration matrix

We recall from Chapter 3, that the LDC iteration can be written in matrix notation as (cf. (3.23))

\[
\begin{pmatrix}
L_h^l & 0 & B_{l,r}^{h,H} & 0 \\
0 & L_b^l & 0 & 0 \\
B_{l,1}^H & L_{1,1}^H & B_{c,1}^H \\
0 & 0 & B_{c,1}^H & L_c^H
\end{pmatrix}
\begin{pmatrix}
u_h^{l,i} \\
u_b^{l,i} \\
u_{1,1}^{H,i} \\
u_{c,1}^{H,i}
\end{pmatrix}
= 
\begin{pmatrix}
u_h^{l,i} \\
u_b^{l,i} \\
u_{1,1}^{H,i} \\
u_{c,1}^{H,i}
\end{pmatrix}
+ 
\begin{pmatrix}
f_h^{l,i} \\
f_b^{l,i} \\
f_{1,1}^{H,i} \\
f_{c,1}^{H,i}
\end{pmatrix},
\]  
(5.2)

or, in short notation (cf (3.24)):

\[
L_h^{l,i} u_h^{l,i} = S_h^{l,i} u_h^{l,i-1} + f_h^{l,i}.
\]  
(5.3)

If the LDC algorithm converges, then this equation has a fixed point, which we denote by (cf. (3.25))

\[
u_h^{l,i} = 
\begin{pmatrix}
u_h^{l}

\end{pmatrix},
\]  
(5.4)

This fixed point $u_h^{l,i}$ satisfies, by definition, (cf. (3.26))

\[
L_h^{l,i} u_h^{l,i} = S_h^{l,i} u_h^{l,i} + f_h^{l,i}.
\]  
(5.5)

Introducing the iteration error of the LDC method by

\[
e_{l,i}^{h} = u_h^{l,i} - u_h^{l,i},
\]  
(5.6)

and subtracting (5.3) and (5.5), we are led to the following equation for successive iteration errors

\[
L_h^{l,i} e_{l,i}^{h} = S_h^{l,i} e_{l,i}^{h}.
\]  
(5.7)
Note that the convergence behavior of the LDC algorithm does not depend on the source term and Dirichlet boundary condition. Using the definitions of $L_{H,h}$ and $S_{H,h}$, see (5.2), (5.3), we find:

$$
\begin{pmatrix}
L^h_{0} & 0 & \ldots & 0 \\
0 & L^h_{1} & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & L^h_{n}
\end{pmatrix}
\begin{pmatrix}
e^h_{0} \\
e^h_{1} \\
\vdots \\
e^h_{n}
\end{pmatrix}
= \begin{pmatrix}
0 \\
x^H_{1}L^H_{1}R^H_{1,h}e^h_{i-1} + x^H_{i}b^H_{i,r}e^H_{i-1} \\
\vdots \\
\end{pmatrix}.
$$

The first equation of this system yields

$$
e^h_{0} = -(L^h)^{-1}b^H_{i,r}p^H_{i,r}e^H_{i-1}.
$$

Replacing $i$ with $i-1$ in (5.9), we can reformulate the last three equations in system (5.8) as:

$$
\begin{pmatrix}
L^h_{0} & 0 & \ldots & 0 \\
0 & L^h_{1} & \ldots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & L^h_{n}
\end{pmatrix}
\begin{pmatrix}
e^H_{0} \\
e^H_{1} \\
\vdots \\
e^H_{n}
\end{pmatrix}
= \begin{pmatrix}
0 \\
x^H_{1}L^H_{1}R^H_{1,h}e^H_{i-1} - x^H_{i}b^H_{i,r}e^H_{i-1} \\
\vdots \\
\end{pmatrix}.
$$

or, equivalently,

$$
L^H_{1}e^H_{i} = \begin{pmatrix}
1 \\
0 \\
\vdots \\
0
\end{pmatrix}
X^H_{1}\left[B^H_{i,r} - L^H_{1}R^H_{1,h}(L^h)^{-1}b^H_{i,r}p^H_{i,r}\right]e^H_{i-1}.
$$

This leads to the following theorem.

**Theorem 5.1** Consider the following iteration that takes place on the interface only:

$$
e^H_{i+1} = Me^H_{i}, \quad i = 1, 2, \ldots
$$

in which the iteration matrix $M : G(\Gamma^H) \rightarrow G(\Gamma^H)$ is defined by

$$
M = (0 I 0) (L^H)^{-1} \begin{pmatrix}
1 \\
0 \\
0
\end{pmatrix} X^H_{1}\left[B^H_{i,r} - L^H_{1}R^H_{1,h}(L^h)^{-1}b^H_{i,r}p^H_{i,r}\right]e^H_{i-1}.
$$

As before, $e^H_{i+1}$ is the difference of the successive coarse grid approximations $u^H_{i+1}$ on the interface with the fixed point values $u^H_{i}$ on the interface. If the iteration (5.12) converges, then the LDC algorithm converges.

**Proof.** It is easily verified, that (5.11) gives (5.12). Equation (5.12) describes the behavior of component $e^H_{i+1}$ of the iteration error; the other components of the iteration error can
be expressed in $e^i_{H,i}$ by (5.9) and (5.11). Therefore, if we can prove that $e^i_{H,i} \rightarrow 0$ \((i \rightarrow \infty)\), it follows that $e^i_{H,h} \rightarrow 0$ \((i \rightarrow \infty)\), which means that the LDC algorithm converges to a fixed point.

Expression (5.13) for the iteration matrix differs from the expressions by Hackbusch [Hackbusch, 1984, Sec. 3.3.1], and Ferket [Ferket, 1996, Th. 3.9], and is believed to be new. Hackbusch studies the LDC iteration expressed in grid functions defined on $\Gamma^h$, i.e., the set of fine grid points on the interface. Ferket formulates the iteration in terms of the complete composite grid iteration error $e_{H,h}^i$. Our iteration matrix $M$ operates on grid functions defined on $\Gamma^H$, i.e., the set of coarse grid points on the interface.

From Theorem 5.1 we conclude that we can prove convergence of the LDC algorithm by showing that the spectral radius of the iteration matrix $M$ is less than one. It is sufficient to show that the norm of $M$ is less than one in some matrix norm. This will be the topic of the next sections.

### 5.2 Norm of the iteration matrix—continuous local problem

In this section, we estimate the infinity norm of the iteration matrix $M$ from Theorem 5.1 for a model problem with special choices of the grids and discretizations in the LDC algorithm. We will prove convergence results for the Poisson problem on the unit square, viz.

\[
\begin{align*}
\Delta u &= f, & \text{in } \Omega = (0,1)^2, \\
\ u &= g, & \text{on } \partial \Omega.
\end{align*}
\]

We choose the computational grid $\Omega^H$ to be a uniform grid with grid sizes $\Delta x = \Delta y = H$, $N := 1/H$ integer:

\[
\Omega^H = \{(lH,jH) \mid l = 1,2,\ldots,N-1, \ j = 1,2,\ldots,N-1\}.
\]

Furthermore, we let $L^H$ be the standard five-point discretization of the Laplacian on $\Omega^H$. We take the area of high activity $\Omega_1$ to be $\Omega_1 = (0,\gamma)^2$, $0 < \gamma < 1$, with $\gamma$ a multiple of $H$. As before, $\Omega^H_1 = \Omega^H \cap \Omega_1$.

In this section, we will replace $L^h_1$ by $\Delta$, the continuous Laplacian on $\Omega_1$. This corresponds to letting $h \rightarrow 0$. Of course, $L^h$ will normally be a discrete operator, and we will take $L^h_1$ to be discrete in Section 5.3. We choose $L^h_1 = \Delta$ in this section, because this simplifies the analysis significantly whereas the estimate for the norm of the iteration matrix will show the same behavior as the estimate in Section 5.3. For the interpolation operator $P^h$ on the interface, we will use trigonometric interpolation; this is explained in Section 5.2.1. The reason for using this special type of interpolation will become apparent in Section 5.2.2. Finally, we will need to use a safety region in the LDC algorithm to be able to prove convergence. In particular, we will choose

\[
\Omega^H_{\text{def}} = \{(x,y) \in \Omega^H \mid x < \gamma - \epsilon \land y < \gamma - \epsilon\},
\]

52

*Convergence behavior of the LDC method as an iterative process*
5.2 Norm of the iteration matrix—continuous local problem

with $c > 0$, $c$ independent of $H$.

We split the iteration matrix $M$ from Theorem 5.1 according to

$$M = M_1 M_2,$$  \hspace{1cm} (5.17)

where

$$M_1 = (0 \ I \ 0) \ (L^H)^{-1} \begin{pmatrix} I & 0 \\ 0 & 0 \end{pmatrix}, \hspace{1cm} M_2 = X_l^H \left[ B_{l,l}^{H,h} - L_{l}^H R_{l}^{H,h} (L_{l}^H)^{-1} B_{l,l}^{h} p_{h,l}^H \right]. \hspace{1cm} (5.18)$$

Most of the effort will concentrate on establishing an estimate of the norm of $M_2$, because

$$||M||_{\infty} \leq ||M_1||_{\infty} ||M_2||_{\infty} \leq \frac{1}{8} ||M_2||_{\infty}. \hspace{1cm} (5.19)$$

The last inequality is a well-known result for the infinity norm of the discrete Laplacian, see, e.g., [Hackbusch, 1992, Sec. 4.4].

The final result we will establish in the next sections is Theorem 5.7. This theorem states that under the above assumptions, the iteration matrix $M$ from Theorem 5.1 satisfies

$$||M||_{\infty} \leq C H^2, \hspace{1cm} (5.20)$$

with $C$ independent of $H$. This upper bound for the norm of the iteration matrix implies, that the norm of $M$ can become arbitrarily small when the grid size $H$ of the coarse grid tends to zero. Note that this result holds for the LDC algorithm with a safety region only.

5.2.1 Trigonometric interpolation

We assume $p_{h,l}^H$ to be interpolation by a trigonometric polynomial. Consider the interval $(0, \gamma)$ with $\gamma = kH$, $k$ a positive integer, and define the nodes $x_j = jH, j = 0, 1, \ldots, k$. Let function values $g_j, j = 0, 1, \ldots, k$, be given with $g_0 = g_k = 0$. As an interpolating function on $(0, \gamma)$, we will use the trigonometric polynomial $g$, defined by

$$g(x) = \sum_{m=1}^{k-1} \alpha_m \sin \left( \frac{m \pi x}{\gamma} \right), \hspace{1cm} x \in (0, \gamma). \hspace{1cm} (5.21)$$

The right hand side of (5.21) is a finite Fourier sine series. The coefficients $\alpha_m, m = 1, 2, \ldots, k - 1$, follow from the system of equations

$$g(x_j) = g_j = \sum_{m=1}^{k-1} \alpha_m \sin \left( \frac{m \pi x_j}{\gamma} \right), \hspace{1cm} j = 1, 2, \ldots, k - 1. \hspace{1cm} (5.22)$$
Convergence behavior of the LDC method as an iterative process

Introducing the matrix $A$ and vectors $a$, $b$

\[
A = \begin{pmatrix}
\sin \left( \frac{\pi x_1}{\gamma} \right) & \sin \left( \frac{2\pi x_1}{\gamma} \right) & \cdots & \sin \left( \frac{(k-1)\pi x_1}{\gamma} \right) \\
n \sin \left( \frac{\pi x_2}{\gamma} \right) & \sin \left( \frac{2\pi x_2}{\gamma} \right) & \cdots & \sin \left( \frac{(k-1)\pi x_2}{\gamma} \right) \\
\vdots & \vdots & \ddots & \vdots \\
\sin \left( \frac{\pi x_{k-1}}{\gamma} \right) & \sin \left( \frac{2\pi x_{k-1}}{\gamma} \right) & \cdots & \sin \left( \frac{(k-1)\pi x_{k-1}}{\gamma} \right)
\end{pmatrix}, \quad (5.23)
\]

\[
a = (\alpha_1, \alpha_2, \ldots, \alpha_{k-1})^T, \quad (5.24)
\]

\[
b = (g_1, g_2, \ldots, g_{k-1})^T, \quad (5.25)
\]

the system of equations (5.22) can also be written as $Aa = b$. In this way, we find

\[
a = A^{-1}b. \quad (5.26)
\]

The set of functions

\[
\sin \left( \frac{m\pi x}{\gamma} \right), \quad m = 1, 2, \ldots, k-1,
\]

satisfies discrete orthogonality relations, see e.g. [Schwarz, 1986, Satz 4.4, p. 147]. From these relations, it follows that the columns of the matrix $A$ are orthogonal.

**Theorem 5.2** The matrix $A^{-1}$ may be expressed in terms of $A$ as

\[
A^{-1} = \frac{2H}{\gamma} A = \frac{2}{k} A. \quad (5.28)
\]

**Proof.** See [Schwarz, 1986] Ch. 4.

**Corollary 5.3** For the 2-norm and the infinity norm of $A^{-1}$, we have

\[
\|A^{-1}\|_2 = \sqrt{\frac{2}{k}}, \quad \|A^{-1}\|_\infty < \sqrt{2}. \quad (5.29)
\]

**Proof.** The matrix

\[
\tilde{A} := \sqrt{\frac{2}{k}} A,
\]

is symmetric and, by Theorem 5.2, orthogonal. Therefore

\[
\|A^{-1}\|_2 = \sqrt{\frac{2}{k}} \|\tilde{A}^{-1}\|_2 = \sqrt{\frac{2}{k}}.
\]

A straightforward estimate gives

\[
\|A^{-1}\|_\infty \leq \sqrt{k-1} \|A^{-1}\|_2 \leq \sqrt{\frac{k-1}{k}} \sqrt{2} < \sqrt{2}.
\]

This proves the corollary.
5.2 Norm of the iteration matrix—continuous local problem

5.2.2 Norm of the iteration matrix

In this section, we will derive an estimate for the infinity norm of \( M_2 \). Note that \( M_2 : G(\Gamma^H) \to G(\Omega^H) \). To estimate the infinity norm of \( M_2 \), we let \( g^H_l \) be an arbitrary vector in \( G(\Gamma^H) \) and split the vector \( g^H_l \) in

\[
g^H_l = g^H_{\text{hor}} + g^H_{\text{vert}} + g^H_{\text{corner}}. \tag{5.30}
\]

Here, the definitions of \( g^H_{\text{hor}} \), \( g^H_{\text{vert}} \), and \( g^H_{\text{corner}} \) are as one would expect:

\[
g^H_{\text{hor}}(x,y) := \begin{cases} g^H_l(x,y), & (x,y) \in \Gamma^H_{\text{hor}} := \{(lH,\gamma) \mid l = 1,2,\ldots,k-1\}, \\ 0, & (x,y) \in \Gamma^H \setminus \Gamma^H_{\text{hor}} \end{cases}, \tag{5.31}
\]

\[
g^H_{\text{vert}}(x,y) := \begin{cases} g^H_l(x,y), & (x,y) \in \Gamma^H_{\text{vert}} := \{(\gamma,jH) \mid j = 1,2,\ldots,k-1\}, \\ 0, & (x,y) \in \Gamma^H \setminus \Gamma^H_{\text{vert}} \end{cases}, \tag{5.32}
\]

\[
g^H_{\text{corner}}(x,y) := \begin{cases} g^H_l(x,y), & (x,y) = (\gamma,\gamma), \\ 0, & (x,y) \in \Gamma^H \setminus \{(\gamma,\gamma)\}. \tag{5.33}
\]

For the moment, we will assume that both \( g^H_{\text{vert}} \) and \( g^H_{\text{corner}} \) equal zero. Hence, we will only give an estimate of \( \| M_2 g^H_{\text{hor}} \|_\infty \). At the end of this section, we will treat the case of non-zero \( g^H_{\text{vert}} \). Section 5.2.3 deals with the case of non-zero \( g^H_{\text{corner}} \).

From the definition of \( M_2 \), see (5.18), we have

\[
M_2 g^H_{\text{hor}} = X^H_l \left[ B^H_{l,\Gamma} g^H_{\text{hor}} + L^H R^H g^H_{\text{hor}} u_{\text{hor}} \right], \tag{5.34}
\]

where

\[
u_{\text{hor}} = - (L^H)^{-1} B^h_{l,\Gamma} p^h R^H g^H_{\text{hor}}. \tag{5.35}
\]

In view of the definitions of \( L^H \), \( B^h_{l,\Gamma} \), and \( p^h \), it follows that \( u_{\text{hor}} \) is the solution of the boundary value problem

\[
\begin{align*}
\Delta u_{\text{hor}} &= 0, & \text{in } \Omega_1, \\
u_{\text{hor}} &= g_{\text{hor}}, & \text{on } \Gamma, \\
u_{\text{hor}} &= 0, & \text{on } \partial \Omega_1 \setminus \Gamma. \tag{5.36}
\end{align*}
\]

In (5.36), \( g_{\text{hor}} = p^h R^H g^H_{\text{hor}} \) vanishes on the vertical part of the interface \( \Gamma \), i.e., \( g_{\text{hor}}(\gamma,y) = 0 \) for \( y \in (0,\gamma) \). The function \( g_{\text{hor}} \) is the trigonometric polynomial interpolating \( g^H_{\text{hor}} \) on the horizontal part of the interface. Hence, \( g_{\text{hor}}(x,\gamma) \) has the form (5.21) for \( x \in (0,\gamma) \), and its coefficients \( \alpha_m, m = 1,2,\ldots,k-1 \), satisfy (cf. Corollary 5.3)

\[
|\alpha_m| < \sqrt{2} \| g^H_{\text{hor}} \|_\infty, \quad m = 1,2,\ldots,k-1. \tag{5.37}
\]

Due to the specific form of \( g_{\text{hor}} \), the exact solution of boundary value problem (5.36) is readily seen to be

\[
u_{\text{hor}}(x,y) = \sum_{m=1}^{k-1} \alpha_m \sin \left( \frac{m\pi x}{\gamma} \right) \frac{\sinh \left( \frac{m\pi y}{\gamma} \right)}{\sinh(m\pi)}, \quad (x,y) \in \Omega_1. \tag{5.38}
\]
Equation \(5.34\) states, that \(M_2 g^H_{\text{hor}}(x, y)\) equals the residual of the standard five-point stencil for the Laplacian applied to the function \(u_{\text{hor}}\) in \((x, y)\) for all grid points \((x, y) \in \Omega^H_{\text{def}}\). For this reason, we find

\[
(M_2 g^H_{\text{hor}})(x, y) = \frac{1}{H^2} \left[ u_{\text{hor}}(x + H, y) - 2u_{\text{hor}}(x, y) + u_{\text{hor}}(x - H, y) \right. \\
\left. + u_{\text{hor}}(x, y + H) - 2u_{\text{hor}}(x, y) + u_{\text{hor}}(x, y - H) \right].
\]

(5.39)

Using the equalities

\[
\sin \left( \frac{m\pi(x + H)}{\gamma} \right) - 2 \sin \left( \frac{m\pi x}{\gamma} \right) + \sin \left( \frac{m\pi(x - H)}{\gamma} \right) = -4 \sin^2 \left( \frac{m\pi H}{2\gamma} \right) \sin \left( \frac{m\pi x}{\gamma} \right),
\]

(5.40)

\[
\sinh \left( \frac{m\pi(y + H)}{\gamma} \right) - 2 \sinh \left( \frac{m\pi y}{\gamma} \right) + \sinh \left( \frac{m\pi(y - H)}{\gamma} \right) = 4 \sinh^2 \left( \frac{m\pi H}{2\gamma} \right) \sinh \left( \frac{m\pi y}{\gamma} \right),
\]

(5.41)

we find the following expression for \((M_2 g^H_{\text{hor}})(x, y)\):

\[
(M_2 g^H_{\text{hor}})(x, y) = \frac{4}{H^2} \sum_{m=1}^{k-1} \alpha_m \delta_m \sin \left( \frac{m\pi x}{\gamma} \right) \frac{\sinh(m\pi y/\gamma)}{\sinh(m\pi)}.
\]

(5.42)

Here, \(\delta_m\) is defined by

\[
\delta_m = \sinh^2 \left( \frac{m\pi H}{2\gamma} \right) - \sin^2 \left( \frac{m\pi H}{2\gamma} \right).
\]

(5.43)

We give an estimate for \(|\delta_m|\) in the following lemma.

**Lemma 5.4** For \(|\delta_m|\), the following upper bound holds:

\[
|\delta_m| \leq C H^4 \left( \frac{m\pi}{\gamma} \right)^4,
\]

(5.44)

with \(C\) independent of \(H\).

**Proof.** Consider the function \(f\) defined by

\[
f(x) = \sinh^2 x - \sin^2 x.
\]

Developing the function in a Taylor expansion around 0, we find

\[
f(x) = \frac{x^4}{4!} f^{(4)}(\xi), \quad \xi \in (0, x), \quad f^{(4)}(x) = 8(\cosh^2 x + \sinh^2 x - \cos^2 x + \sin^2 x).
\]
Because
\[ \delta_m = \frac{f\left(\frac{m\pi H}{2}\right)}{f^{(4)}(x)} = \frac{1}{4^4} \left(\frac{m\pi H}{2}\right)^4 f^{(4)}(x), \quad x \in (0, \pi/2), \]
and for all \( x \in (0, \pi/2) \)
\[ |f^{(4)}(x)| \leq 8 \left( \cosh^2(\pi/2) + \sinh^2(\pi/2) + 1 \right), \]
we find the following upper bound for \( |\delta_m| \)
\[ |\delta_m| \leq H^4 \frac{4^4}{48} \left(\frac{m\pi}{2}\right)^4 \left( \cosh^2(\pi/2) + \sinh^2(\pi/2) + 1 \right) = CH^4 \left(\frac{m\pi}{2}\right)^4, \]
with \( C \) independent of \( H \). \( \square \)

From (5.42), we find by (5.37) and Lemma 5.4:
\[ \left| (M^2 g^H_{\text{hor}})(x, y) \right| \leq 4CH^2 \sqrt{2} \|g^H_{\text{hor}}\|_\infty \sum_{m=1}^{\infty} \left(\frac{m\pi}{2}\right)^4 \frac{\sinh(m\pi y/\gamma)}{\sinh(m\pi)}. \quad (5.45) \]

Note that this upper bound depends on \( H, \|g^H_{\text{hor}}\|_\infty, \gamma, \) and \( y \) only (the upper limit of the sum, \( k - 1 \), is a function of \( y \) and \( H \), because \( k = \gamma/H \)). We will show that we can give an upper bound for the sum in the right hand side of (5.45) which does not depend on \( H \). This implies, that
\[ \left\| M^2 g^H_{\text{hor}} \right\|_\infty \leq C_1 H^2 \|g^H_{\text{hor}}\|_\infty, \quad (5.46) \]
in which \( C_1 \) is a constant independent of \( H \).

To establish the \( H \)-independent upper bound for the sum, we use the assumption made at the start of this section, that \( y < \gamma - \epsilon \) with \( \epsilon \) independent of \( H \). For the sum in (5.45), we find
\[ \sum_{m=1}^{\infty} \left(\frac{m\pi}{2}\right)^4 \frac{\sinh(m\pi y/\gamma)}{\sinh(m\pi)} \leq \sum_{m=1}^{\infty} \left(\frac{m\pi}{\gamma}\right)^4 \frac{\sinh(m\pi y/\gamma)}{\sinh(m\pi)} < \infty. \quad (5.47) \]

Convergence of the series in (5.47) can be shown using d’Alembert’s theorem. Consider the series
\[ \sum_{m=1}^{\infty} a_m, \quad a_m = \left(\frac{m\pi}{\gamma}\right)^4 \frac{\sinh(m\pi y/\gamma)}{\sinh(m\pi)}. \quad (5.48) \]

We have
\[ \frac{\sinh(m\pi y/\gamma)}{\sinh(m\pi)} = \frac{\exp(m\pi y/\gamma) - \exp(-m\pi y/\gamma)}{\exp(m\pi) - \exp(-m\pi)} = \frac{\exp(m\pi(y/\gamma - 1))}{1 - \exp(-2m\pi/\gamma)}, \quad (5.49) \]
so that
\[
\frac{a_{m+1}}{a_m} = \left(\frac{m+1}{m}\right)^4 \exp\left(\pi(y / \gamma - 1)\right) \frac{1 - \exp(-2(m+1)\pi y / \gamma)}{1 - \exp(-2m\pi y / \gamma)} \frac{1 - \exp(-2m\pi)}{1 - \exp(-2(m+1)\pi)} \to \exp\left(\pi\left(\frac{y}{\gamma} - 1\right)\right), \quad m \to \infty.
\]

(5.50)
For \((x, y) \in \Omega_{\text{def}}^H\), we have \(y < \gamma - \epsilon\), and we find
\[
\exp\left(\pi\left(\frac{y}{\gamma} - 1\right)\right) < \exp\left(-\frac{\pi\epsilon}{\gamma}\right) < 1,
\]
which proves the convergence claimed in (5.47).

We have now shown that for all \((x, y) \in \Omega_{\text{def}}^H\)
\[
\|M_2g_{\text{hor}}^H(x, y)\| \leq C_1H^2\|g_{\text{hor}}^\Gamma\|_{\infty},
\]
with \(C_1\) independent of \(H\).

The results of this section are summarized in the following lemma.

**Lemma 5.5** Consider the LDC algorithm for the Poisson problem (5.14) with the following settings. Let \(L^H\) be the standard five-point discretization of the Laplacian on a uniform grid with grid sizes \(\Delta x = \Delta y = H\) as in (5.15). Let \(\Omega_l = (0, \gamma)^2\) with \(0 < \gamma < 1\) and \(\gamma\) a multiple of \(H\). Let \(L^H_l\) be the continuous Laplacian on \(\Omega_l\). Let \(P^h,^H\) be interpolation by a trigonometric polynomial. Finally, let \(\Omega_{\text{def}}^H = \{(x, y) \in \Omega^H_l| x < \gamma - \epsilon \land y < \gamma - \epsilon\}\),

(5.53)
for some \(\epsilon > 0\), \(\epsilon\) independent of \(H\). Then
\[
\|M_2g_{\text{hor}}^H\|_{\infty} \leq C_1H^2\|g_{\text{hor}}^\Gamma\|_{\infty},
\]

(5.54)
for all \(g_{\text{hor}}^H \in G(\Gamma^H)\) with \(g_{\text{hor}}^H(x, y) \neq 0\) for \((x, y) \in \Gamma_{\text{hor}}^H\) only. In (5.54), \(C_1\) is independent of \(H\).

Lemma 5.5 is the largest part of establishing the upper bound for the norm of the iteration matrix \(M\) of the LDC algorithm. The case of non-zero artificial Dirichlet boundary conditions on the vertical part of the interface is similar. Indeed, starting from (5.30) we may also assume that \(g_{\text{hor}}^H\) and \(g_{\text{corner}}^H\) equal zero. Analogous to what we have seen above, we can prescribe artificial Dirichlet boundary conditions \(g_{\text{vert}}^H = P^h,^Hg_{\text{vert}}^H\) by using interpolation with a trigonometric polynomial on the vertical part of the interface. The function \(g_{\text{vert}}^H\) vanishes on the horizontal part of the interface. This leads to a local boundary value problem similar to (5.36). Hence, an analogue of Lemma 5.5 holds.

**Lemma 5.6** Consider the LDC algorithm for the Poisson problem (5.14) with the following settings. Let \(L^H\) be the standard five-point discretization of the Laplacian on a uniform grid with grid sizes \(\Delta x = \Delta y = H\) as in (5.15). Let \(\Omega_l = (0, \gamma)^2\) with \(0 < \gamma < 1\) and \(\gamma\) a multiple
5.2 Norm of the iteration matrix—continuous local problem

of \( H \). Let \( L_h^{1} \) be the continuous Laplacian on \( \Omega_1 \). Let \( P^{h,1} \) be interpolation by a trigonometric polynomial. Finally, let

\[
\Omega_{d,1}^{1} = \{ (x, y) \in \Omega_1^{1} \mid x < \gamma - \epsilon \land y < \gamma - \epsilon \},
\]

(5.55)

for some \( \epsilon > 0 \), \( \epsilon \) independent of \( H \). Then

\[
\| M^2 g^{1} \|_{\infty} \leq C_2 H^2 \| g^{1} \|_{\infty},
\]

(5.56)

for all \( g^{1} \in G(\Gamma^1) \) with \( g^{1}(x, y) \neq 0 \) for \( (x, y) \in \Gamma_{\text{vert}}^1 \) only. In (5.56), \( C_2 \) is independent of \( H \).

Note that the use of a safety region corresponds to the concept of interior regularity used by Hackbusch [Hackbusch, 1984] and Wappler [Wappler, 1999].

5.2.3 Treatment of a non-zero value at the corner of the interface

In the previous section, we used trigonometric interpolation on the interface. However, so far we have assumed that the artificial Dirichlet boundary conditions equal zero at the corner of the interface, and hence we have defined the interpolation operator \( P^{h,1} \) for vectors \( g^{1} \in G(\Gamma^1) \) with \( g^{1}(\gamma, \gamma) \) equal to zero only.

If \( g^{1}(\gamma, \gamma) \neq 0 \), we proceed as follows. The idea is to subtract a suitable grid function to accomplish a zero corner value. Consider the function

\[
v(x, y) = \frac{xy}{\gamma^2}, \quad (x, y) \in \Omega_1.
\]

(5.57)

The function \( v \) is linear on the horizontal and the vertical part of the interface \( \Gamma \), and \( v(\gamma, \gamma) = 1 \). Furthermore, \( v \) vanishes on \( \partial \Omega_1 \setminus \Gamma \), and \( v \) is harmonic in \( \Omega_1 \). Also, the projection of \( v \) on \( \Omega^1 \) is discretely harmonic. For \( g^{1} \in G(\Gamma^1) \) with \( g^{1}(\gamma, \gamma) \neq 0 \), we introduce \( \tilde{g}^{1} \in G(\Gamma^1) \) by

\[
\tilde{g}^{1} = g^{1} - g^{1}(\gamma, \gamma) \cdot v|_{\Gamma^1}.
\]

(5.58)

The grid function \( \tilde{g}^{1} \) is zero in the corner \( (\gamma, \gamma) \) of the interface. Hence, we can split \( \tilde{g}^{1} \) according to, cf. (5.30),

\[
\tilde{g}^{1} = \tilde{g}^{1}_{\text{hor}} + \tilde{g}^{1}_{\text{vert}},
\]

(5.59)

and use trigonometric interpolation on both the horizontal and the vertical part of the interface to find artificial Dirichlet boundary conditions \( \tilde{g}^{1}_{\text{hor}}, \tilde{g}^{1}_{\text{vert}} \), respectively. We use the following definition for the interpolation operator \( P^{h,1} \):

\[
P^{h,1} g^{1} = \tilde{g}^{1}_{\text{hor}} + \tilde{g}^{1}_{\text{vert}} + g^{1}(\gamma, \gamma) \cdot v|_{\Gamma^1}.
\]

(5.60)

Using this definition, we have

\[
M_2 g^{1} = M_2 \tilde{g}^{1}_{\text{hor}} + M_2 \tilde{g}^{1}_{\text{vert}} + g^{1}(\gamma, \gamma) \cdot M_2 (v|_{\Gamma^1}) = M_2 \tilde{g}^{1}_{\text{hor}} + M_2 \tilde{g}^{1}_{\text{vert}}.
\]

(5.61)
Here, we have used

$$M_2(v_{\Gamma H}) = X_H^H \left[ B_{l,r}^H (v_{\Gamma H}) - L_{l,r}^H R_{r}^{H,H} (L_{l,r}^H)^{-1} B_{l,r}^H (v_{\Gamma H}) \right]$$

$$= X_H^H \left[ B_{l,r}^H (v_{\Gamma H}) - L_{l,r}^H R_{r}^{H,H} (L_{l,r}^H)^{-1} B_{l,r}^H (v_{\Gamma r}) \right]$$

$$= X_H^H \left[ B_{l,r}^H (v_{\Gamma H}) + L_{l,r}^H (v_{\Gamma \Omega H}) \right] = 0,$$

(5.62)

because the projection of $v$ is discretely harmonic on the local coarse grid $\Omega_l^H$. For the remaining two terms in the right hand side of (5.61), we have the results from the previous section. This leads to the main theorem.

**Theorem 5.7** Consider the LDC algorithm for the Poisson problem (5.14) with the following settings. Let $L^H$ be the standard five-point discretization of the Laplacian on a uniform grid with grid sizes $\Delta x = \Delta y = H$ as in (5.15). Let $\Omega_l = (0, \gamma)^2$ with $0 < \gamma < 1$ and $\gamma$ a multiple of $H$. Let $L^h_l$ be the continuous Laplacian on $\Omega_l$. Let $P^{h,H}$ be trigonometric interpolation. Finally, let $\Omega_{l,0}^H = \{ (x,y) \in \Omega^H_l \mid x < \gamma - \epsilon \wedge y < \gamma - \epsilon \},$

(5.63)

for some $\epsilon > 0$, $\epsilon$ independent of $H$. Then the following upper bound for the norm of the iteration matrix $M$ of the LDC algorithm holds:

$$\|M\|_{\infty} \leq C H^2,$$

(5.64)

with $C$ independent of $H$.

**Proof.** Let $g^H_{\Gamma} \in G(\Gamma^H)$. Define $\tilde{g}^H_{\Gamma} \in G(\Gamma^H)$ as in (5.58). Using (5.61) and Lemmas 5.5 and 5.6 we find

$$\|M_2 g^H_{\Gamma}\|_{\infty} = \|M_2 \tilde{g}^H_{\Gamma,0} + M_2 \tilde{g}^H_{\Gamma,\text{cert}}\|_{\infty} \leq (C_1 + C_2) H^2 \|\tilde{g}^H_{\Gamma}\|_{\infty},$$

with $C_1, C_2$ independent of $H$. From the definition of $\tilde{g}^H_{\Gamma}$, it follows that $\|\tilde{g}^H_{\Gamma}\|_{\infty} \leq 2 \|g^H_{\Gamma}\|_{\infty}$, and hence

$$\|M_2\|_{\infty} \leq 2(C_1 + C_2) H^2.$$

Using (5.19), we conclude that

$$\|M\|_{\infty} \leq \frac{1}{4} (C_1 + C_2) H^2 = CH^2,$$

with $C$ independent of $H$. 

Although our proofs may seem to rely quite heavily on the use of trigonometric interpolation on the interface, we expect similar results to hold if we use linear interpolation in the LDC iteration. We may motivate this claim as follows: rather than using any special properties of trigonometric interpolation in our proofs, this interpolation turns out to be very convenient when solving boundary value problem (5.36). Due to the fact that we have used trigonometric interpolation rather than linear or quadratic interpolation, we can readily write down the analytical solution of (5.36), cf. (5.38). The influence of
the type of interpolation used is further investigated in some numerical experiments in Section 5.4.

We would also like to remark at this point, that the use of a safety region is essential for our convergence proof. The asymptotic behavior of Theorem 5.7 will not occur if the assumption that the width \( \epsilon \) of the safety region is independent of \( H \) does not hold. This will be shown in numerical experiments in Section 5.4. In practice, the convergence is however still very fast, even when \( \epsilon = 0 \). For this reason, usage of a safety region does not seem to be necessary in practice, and we will set \( \epsilon = 0 \) in Chapters 4–7.

5.3 Norm of the iteration matrix—discrete local problem

In Section 5.2, we derived an \( O(H^2) \) upper bound for the iteration matrix of the LDC algorithm with a safety region. One of the assumptions in Section 5.2 was that the local problem was a continuous boundary value problem. In other words, we assumed \( h \) to be equal to zero. In this section, we will drop this assumption and give an upper bound for the iteration matrix for positive values of \( h \).

Hence, we consider the same boundary value problem (5.14) as in Section 5.2. As before, we choose the computational grid \( \Omega^H \) to be a uniform grid with grid sizes \( \Delta x = \Delta y = H \), \( N := 1/H \) integer, as in (5.15), and we let \( L^H \) be the standard five-point discretization of the Laplacian on \( \Omega^H \). We take the area of high activity \( \Omega_1 \) to be \( \Omega_1 = (0, \gamma)^2 \), \( 0 < \gamma < 1 \), with \( \gamma \) a multiple of \( H \).

In this section, we choose a uniform grid \( \Omega^h_1 \) in \( \Omega_1 \) with grid size \( \Delta x = \Delta y = h \), \( n := \gamma/h \) integer:

\[
\Omega^h_1 = \{(lh, jh) \mid l = 1, 2, \ldots, n - 1, j = 1, 2, \ldots, n - 1\}.
\] (5.65)

We assume the refinement factor \( \sigma := H/h \) to be integer. We let \( L^h \) be the standard five-point discretization of the Laplacian on \( \Omega^h_1 \). For the interpolation operator \( P^{h,H} \) on the interface, we will use trigonometric interpolation as before. Like in Section 5.2, we will need to use a safety region in the LDC algorithm to be able to prove convergence, and we will choose \( \Omega^H_{\text{def}} \) as in (5.16). Again, the analysis will focus on the matrix \( M_2 \), cf. (5.18), (5.19).

The final result we will establish in the next sections is Theorem 5.12. This theorem states that under the above assumptions, the iteration matrix \( M \) from Theorem 5.1 satisfies

\[
\|M\|_\infty \leq C H^2,
\] (5.66)

with \( C \) independent of \( H \). This theorem is the discrete analogon of Theorem 5.7, the main result of Section 5.2. Note that this result holds for the LDC algorithm with a safety region only.
5.3.1 Norm of the iteration matrix

As in Section 5.2.2, we assume $g^H \in G(\Gamma^H)$ to be given, and we first consider the case where both $g_{\mathrm{vert}}^T$ and $g_{\mathrm{corner}}^T$ equal zero, cf. (5.30). Hence, we will only give an estimate of $\|M_2 g^H\|_\infty$. At the end of this section, we will treat the case of non-zero $g_{\mathrm{vert}}^T$. Section 5.3.2 deals with the case of non-zero $g_{\mathrm{corner}}^T$. As before, we use trigonometric interpolation to find artificial Dirichlet boundary conditions $g_{\mathrm{hor}}^h$ on the horizontal part of the interface.

From the definition of $M_2$, see (5.18), we have

$$M_2 g^H = X^H \left[ B^H_{l,\Gamma} g_{\mathrm{hor}}^H + L^H_{l,\Gamma,h} u_{\mathrm{hor}}^h \right],$$

(5.67)

where

$$u_{\mathrm{hor}}^h = - \left( L^H_{l,\Gamma} \right)^{-1} B^H_{l,\Gamma} p_{\mathrm{hor}}^H g_{\mathrm{hor}}^H.$$  

(5.68)

In view of the definitions of $L^H_{l,\Gamma}$, $B^H_{l,\Gamma}$, and $p_{\mathrm{hor}}^H$, it follows that $u_{\mathrm{hor}}^h$ satisfies a discrete equivalent of (5.36); $u_{\mathrm{hor}}^h$ satisfies the discrete Laplace equation, viz.

$$u_{\mathrm{hor}}^h(x + h, y) - 2u_{\mathrm{hor}}^h(x, y) + u_{\mathrm{hor}}^h(x - h, y)$$
$$+ u_{\mathrm{hor}}^h(x, y + h) - 2u_{\mathrm{hor}}^h(x, y) + u_{\mathrm{hor}}^h(x, y - h) = 0,$$

(5.69)

in each grid point $(x, y) \in \Omega^H_l$. Artificial Dirichlet boundary conditions $p_{\mathrm{hor}}^H g_{\mathrm{hor}}^H$ are prescribed on the interface $\Gamma$; on the horizontal part of $\Gamma$, trigonometric interpolation is used, on the vertical part of $\Gamma$, the artificial Dirichlet boundary conditions vanish. Dirichlet conditions on $\partial \Omega_l \setminus \Gamma$ vanish, too.

Just as for the continuous boundary value problem, we try the separation of variables technique to find $u_{\mathrm{hor}}^h$. So, we set

$$u_{\mathrm{hor}}^h(lh, jh) = X_l Y_j, \quad l, j = 0, 1, \ldots, n.$$  

(5.70)

Substitution into (5.69) and division by $X_l Y_j$ yields

$$\frac{X_{l+1} - 2X_l + X_{l-1}}{X_l} + \frac{Y_{j+1} - 2Y_j + Y_{j-1}}{Y_j} = 0.$$  

(5.71)

We conclude that

$$\frac{X_{l+1} - 2X_l + X_{l-1}}{X_l} = - \frac{Y_{j+1} - 2Y_j + Y_{j-1}}{Y_j} = \mu,$$

(5.72)

with $\mu$, the separation constant, independent of $l$ and $j$. Standard techniques for solving the recurrence relation for $X_l$, viz.

$$X_{l+1} - (2 + \mu)X_l + X_{l-1} = 0,$$

(5.73)

with boundary conditions $X_0 = X_n = 0$ show, that non-trivial solutions $X_l$ exist for

$$\mu_m = -4 \sin^2 \left( \frac{m\pi}{2n} \right),$$  

(5.74)
5.3 Norm of the iteration matrix—discrete local problem

in which \( m = 1, 2, \ldots, n - 1 \), with corresponding solutions

\[
X_l = \sin \left( \frac{m\pi(lh)}{\gamma} \right), \quad l = 0, 1, \ldots, n. \tag{5.75}
\]

Note that these solutions are simply the projections of the continuous solutions on the grid, cf. (5.38). In a similar way, the recurrence relation for \( Y_j \), viz.

\[
Y_{j+1} - (2 - \mu)Y_j + Y_{j-1} = 0, \tag{5.76}
\]

with boundary conditions \( Y_0 = 0, Y_n = 1 \), has the solutions

\[
Y_j = \frac{\sinh(\beta_m j/n)}{\sinh \beta_m} = \frac{\sinh(\beta_m (jh)/\gamma)}{\sinh \beta_m}, \quad j = 0, 1, \ldots, n, \tag{5.77}
\]

in which the coefficients \( \beta_m \) are defined by

\[
\beta_m := 2n \text{Arsinh} \left( \sin \left( \frac{m\pi}{2n} \right) \right) = \frac{2\gamma}{h} \text{Arsinh} \left( \sin \left( \frac{m\pi h}{2\gamma} \right) \right). \tag{5.78}
\]

The solutions (5.75) and (5.77) can be verified by the reader by simple substitution. Notice the similarity between (5.77) and the solutions \( Y(y) = \sinh(m\pi y/\gamma)/\sinh(m\pi) \), \( y \in (0, \gamma) \), of the continuous problem, cf. (5.38), especially as we have

\[
\lim_{h \to 0} \beta_m = \lim_{h \to 0} \frac{2\gamma}{h} \text{Arsinh} \left( \sin \left( \frac{m\pi h}{2\gamma} \right) \right) = m\pi, \tag{5.79}
\]

so that the discrete solutions converge to the continuous solutions for \( h \to 0 \).

Summarizing, the separation of variables method produces the solution

\[
u_{\text{hor}}^h(lh, jh) = \sum_{m=1}^{k-1} \alpha_m \sin \left( \frac{m\pi(lh)}{\gamma} \right) \frac{\sinh(\beta_m (jh)/\gamma)}{\sinh \beta_m}, \tag{5.80}
\]

in which the coefficients \( \alpha_m \) follow from the boundary conditions \( g_{\text{hor}}^h \) on the horizontal part of \( \Gamma^h \). For \( h \to 0 \), (5.80) reduces to the solution (5.38) of the continuous boundary value problem (5.36).

Note that \( u_{\text{hor}}^h = - (L_1^h)^{-1} B_{1, l}^h p^{h, i} g_{\text{hor}}^H \) and hence

\[
(M_2 g_{\text{hor}}^H)(x, y) = (B_{1, l}^H g_{\text{hor}}^H + L_1^H R_{l, h}^H u_{\text{hor}}^h)(x, y) \tag{5.81}
\]

equals the residual of the standard five-point stencil for the Laplacian applied to \( u_{\text{hor}}^h \) in the (coarse) grid point \((x, y) \in \Omega_{\text{def}}^H \). In a similar way as in Section 5.2.2, cf. (5.39), (5.42), we find the following expression for \((M_2 g_{\text{hor}}^H)(x, y)\):

\[
(M_2 g_{\text{hor}}^H)(x, y) = \frac{4}{H^2} \sum_{m=1}^{k-1} \alpha_m \delta_m \sin \left( \frac{m\pi x}{\gamma} \right) \frac{\sinh(\beta_m y/\gamma)}{\sinh \beta_m}. \tag{5.82}
\]
Here, $\delta_m$ is defined by
\[
\delta_m = \sinh^2 \left( \frac{\beta_m H}{2\gamma} \right) - \sin^2 \left( \frac{m\pi H}{2\gamma} \right).
\] (5.83)

We give an estimate for $|\delta_m|$ in the following lemma. This lemma is the discrete equivalent of Lemma 5.4.

**Lemma 5.8** For $|\delta_m|$, the following upper bound holds:
\[
|\delta_m| \leq \left[ C_1 H^2 + D_1 h^2 \right] H^2 \left( \frac{m\pi}{\gamma} \right)^4,
\] (5.84)
with $C_1, D_1$, independent of $H$ and $h$.

**Proof.** To prove the lemma, we will use some Taylor expansions. From (5.78), we have
\[
\beta_m = \frac{2\gamma}{h} f_1 \left( \frac{m\pi h}{2\gamma} \right), \quad f_1(x) = \text{Arsinh}(\sin x).
\]

A simple first order expansion gives
\[
\beta_m = \frac{2\gamma}{h} \frac{m\pi h}{2\gamma} f'_1(\xi) = m\pi f'_1(\xi),
\]
with $\xi \in (0, m\pi h/(2\gamma)) \subset (0, \pi/2)$. Because
\[
f'_1(x) = \frac{\cos x}{\sqrt{1 + \sin^2 x}},
\]
we have $f'_1(\xi) \in (0, 1)$, and hence
\[
0 \leq \beta_m \leq m\pi, \quad m = 1, 2, \ldots, k - 1. \tag{5.85}
\]

Expanding $f_1$ up to third order, we find
\[
f_1(x) = x + \frac{x^3}{3!} f_1^{(3)}(\xi), \quad \xi \in (0, x), \quad f_1^{(3)}(x) = -\frac{4\sqrt{2}(\cos x + \cos(3x))}{(3 - \cos(2x))^{5/2}},
\]
and
\[
\beta_m = \frac{2\gamma}{h} \left[ \frac{m\pi h}{2\gamma} + \frac{1}{3!} \left( \frac{m\pi h}{2\gamma} \right)^3 f_1^{(3)}(\xi_1) \right] = m\pi + \frac{\gamma}{24} \left( \frac{m\pi}{\gamma} \right)^3 h^2 f_1^{(3)}(\xi_1). \tag{5.86}
\]

Next, we define
\[
f_2(x) = \sinh^2 x,
\]
with Taylor expansion
\[
f_2(x) = x^2 + \frac{x^4}{4!} f_2^{(4)}(\xi), \quad \xi \in (0, x), \quad f_2^{(4)}(x) = 8(\cosh^2 x + \sinh^2 x).
\]
Substitution of (5.86) in the expansion of \( f_2 \) yields
\[
\sinh^2 \left( \frac{\beta_m H}{2\gamma} \right) = \left( \frac{\beta_m H}{2\gamma} \right)^2 + \frac{1}{4!} \left( \frac{\beta_m H}{2\gamma} \right)^4 f_2^{(4)}(\xi_2) \\
= \frac{H^2}{4} \left( \frac{m\pi}{\gamma} \right)^2 + \frac{H^4}{384} \left( \frac{\beta_m}{2\gamma} \right)^4 f_2^{(4)}(\xi_2) \\
+ \frac{H^2h^2}{48} \left( \frac{m\pi}{\gamma} \right)^4 f_1^{(3)}(\xi_1) + \frac{H^2h^4}{2304} \left( \frac{m\pi}{\gamma} \right)^6 \left( f_1^{(3)}(\xi_1) \right)^2.
\]
(5.87)

Finally, we introduce the function
\[
f_3(x) = \sin^2 x,
\]
which has the Taylor expansion
\[
f_3(x) = x^2 + \frac{x^4}{4!} f_3^{(4)}(\xi), \quad \xi \in (0, x), \quad f_3^{(4)}(x) = 8(\sin^2 x - \cos^2 x).
\]
Using this expansion, we find
\[
\sin^2 \left( \frac{m\pi H}{2\gamma} \right) = \frac{H^2}{4} \left( \frac{m\pi}{\gamma} \right)^2 + \frac{H^4}{384} \left( \frac{m\pi}{\gamma} \right)^4 f_3^{(4)}(\xi_3).
\]
(5.88)

Combining (5.87) and (5.88), we find the following expression for \( \delta_m \):
\[
\delta_m = \frac{H^4}{384} \left( \frac{\beta_m}{\gamma} \right)^4 f_2^{(4)}(\xi_2) - \frac{H^4}{384} \left( \frac{m\pi}{\gamma} \right)^4 f_3^{(4)}(\xi_3) \\
+ \frac{H^2h^2}{48} \left( \frac{m\pi}{\gamma} \right)^4 f_1^{(3)}(\xi_1) + \frac{H^2h^4}{2304} \left( \frac{m\pi}{\gamma} \right)^6 \left( f_1^{(3)}(\xi_1) \right)^2.
\]

Note that \( \xi_1, \xi_2, \xi_3 \in (0, \pi/2) \), and
\[
|f_1^{(3)}(x)| \leq 2, \quad |f_2^{(4)}(x)| \leq 8(\cosh^2(\pi/2) + \sinh^2(\pi/2)), \quad |f_3^{(4)}(x)| \leq 8,
\]
for all \( x \in (0, \pi/2) \). Setting \( C := 8(\cosh^2(\pi/2) + \sinh^2(\pi/2)) \) and using (5.85), we have
\[
|\delta_m| \leq \left( C + \frac{8H^4}{384} \right) \left( \frac{m\pi}{\gamma} \right)^4 + \frac{H^2h^4}{24} \left( \frac{m\pi}{\gamma} \right)^4 + \frac{H^2h^4}{576} \left( \frac{m\pi}{\gamma} \right)^6.
\]

Because \( mH \leq \gamma \), we have
\[
H^2h^4 \left( \frac{m\pi}{\gamma} \right)^6 = H^2h^2 \left( \frac{m\pi}{\gamma} \right)^4 \left( \frac{m\pi H}{\gamma} \right)^2 \frac{1}{\sigma^2} \leq H^2h^2 \left( \frac{m\pi}{\gamma} \right)^4 \left( \frac{\pi}{\sigma} \right)^2,
\]
and we can incorporate the third term in the second one; this results in
\[
|\delta_m| \leq \left[ C_1 H^2 + D_1 h^2 \right] H^2 \left( \frac{m\pi}{\gamma} \right)^4,
\]
Convergence behavior of the LDC method as an iterative process

with $C_1$, $D_1$, independent of $H$ and $h$.

From (5.82), we find by the bound (5.37) on the coefficients $\alpha_m$ and Lemma 5.8

$$\left| (M_2 g_{\text{hor}}^H (x, y)) \right| \leq 4 \sqrt{2} \left[ C_1 H^2 + D_1 h^2 \right] \| g_{\text{hor}}^H \|_{\infty} \sum_{m=1}^{k-1} \left( \frac{m \pi}{Y} \right)^4 \frac{\sinh(\beta_m y / \gamma)}{\sinh \beta_m}. \quad (5.89)$$

Equation (5.89) is an upper bound that depends on $H$, $h$, $\gamma$, and $y$ (the upper limit of the sum, $k-1$, is a function of $\gamma$ and $H$, because $k = \gamma / H$). For $h \to 0$, one can show that (5.89) reduces to (5.45).

We will show that we can give an upper bound for the sum in the right hand side of (5.89) which does not depend on $H$ or $h$. This implies, that $\left| (M_2 g_{\text{hor}}^H (x, y)) \right|$ is $O(H^2) + O(h^2)$. To establish the grid size independent upper bound for the sum, the analysis is more subtle than in Section 5.2.2, as we now have terms with $\sinh(\beta_m y / \gamma) / \sinh \beta_m$ in the sum, whereas we had terms with $\sinh((m \pi y / \gamma) / \sinh(m \pi)$ in Section 5.2.2. As in Section 5.2.2, we assume that the value of $y$ is smaller than $\gamma$ and does not depend on $H$. This implies, that we consider LDC with a safety region.

We first prove the following lemma.

**Lemma 5.9** For $\beta_m$, $m = 1, 2, \ldots, k-1$, as defined by (5.78), we have

$$\beta_{m+1} \geq \beta_m + \frac{1}{3} \pi \sqrt{3}. \quad (5.90)$$

**Proof.** Define the function $f$ by

$$f(x) = \text{Arsinh}(\sin x).$$

Then

$$f'(x) = \frac{\cos x}{\sqrt{1 + \sin^2 x}},$$

and hence

$$\beta_{m+1} = \frac{2 \gamma}{h} f \left( \frac{(m+1) \pi h}{2 \gamma} \right) = \frac{2 \gamma}{h} \left[ f \left( \frac{m \pi h}{2 \gamma} \right) + \frac{\pi h}{2 \gamma} \cdot f'(\xi) \right] = \beta_m + f'(\xi). \quad (5.91)$$

In (5.91), $\xi \in (m \pi h/(2 \gamma), (m + 1) \pi h/(2 \gamma))$. Because $1 \leq m \leq k-1$, we have $\xi \in (0, \pi/(2 \sigma))$. If $\sigma = 1$, we have the uninteresting situation that $h = H$ in the LDC algorithm. It is easy to show that the iteration matrix equals the zero matrix in this case. Therefore, we assume $\sigma \geq 2$. This implies that $\xi \in (0, \pi/4)$. For these values of $\xi$, we have $f'(\xi) \geq \sqrt{3}/3$. With (5.91), this proves the lemma.

We have

$$\lim_{h \to 0} \beta_1 = \pi, \quad (5.92)$$
and hence $\beta_1$ is larger than, say, $\pi/2$ for $h$ small enough. We introduce the minorizing sequence $\tilde{\beta}_m$ by

$$\tilde{\beta}_1 = \frac{\pi}{2},$$

(5.93)

$$\tilde{\beta}_{m+1} = \tilde{\beta}_m + \frac{1}{3}\pi\sqrt{3}, \quad m = 1, 2, \ldots$$

(5.94)

Due to Lemma 5.9 and the definition of the sequence $\tilde{\beta}_m$, $m = 1, 2, \ldots$, we have

$$\tilde{\beta}_m \geq \beta_m,$$

and hence

$$k \sum_{m=1}^{k-1} \left( \frac{m\pi}{\gamma} \right)^4 \frac{\sinh(\beta_m y/\gamma)}{\sinh \beta_m} \leq \sum_{m=1}^{k-1} \left( \frac{m\pi}{\gamma} \right)^4 \frac{\sinh(\tilde{\beta}_m y/\gamma)}{\sinh \tilde{\beta}_m} \leq \sum_{m=1}^{\infty} \left( \frac{m\pi}{\gamma} \right)^4 \frac{\sinh(\tilde{\beta}_m y/\gamma)}{\sinh \tilde{\beta}_m}$$

$$\leq \infty.$$  

(5.95)

The first inequality holds, because the function

$$f_a(z) = \frac{\sinh(az)}{\sinh z},$$

(5.96)

is decreasing in $z$ for $0 < a < 1$ and $z > 0$. Convergence of the infinite series can again be shown by applying d’Alembert’s theorem as in Section 5.2.2. Define

$$\tilde{a}_m = \left( \frac{m\pi}{\gamma} \right)^4 \frac{\sinh(\tilde{\beta}_m y/\gamma)}{\sinh \tilde{\beta}_m}.$$  

(5.97)

We have

$$\frac{\sinh(\tilde{\beta}_m y/\gamma)}{\sinh \tilde{\beta}_m} = \exp \left( \tilde{\beta}_m \left( \frac{y}{\gamma} - 1 \right) \right) \frac{1 - \exp(-2\tilde{\beta}_m y/\gamma)}{1 - \exp(-2\tilde{\beta}_m)},$$

(5.98)

and therefore

$$\frac{\tilde{a}_{m+1}}{\tilde{a}_m} = \left( \frac{m + 1}{m} \right)^4 \frac{\exp(\tilde{\beta}_m(y/\gamma - 1))}{\exp(\tilde{\beta}_m(y/\gamma - 1))} \frac{1 - \exp(-2\tilde{\beta}_{m+1} y/\gamma)}{1 - \exp(-2\tilde{\beta}_{m+1})} \frac{1 - \exp(-2\tilde{\beta}_m)}{1 - \exp(-2\tilde{\beta}_m)}$$

$$\rightarrow \exp \left( \frac{1}{3}\pi\sqrt{3} \left( \frac{y}{\gamma} - 1 \right) \right), \quad m \rightarrow \infty.$$  

(5.99)

Because

$$\exp \left( \frac{1}{3}\pi\sqrt{3} \left( \frac{y}{\gamma} - 1 \right) \right) < 1,$$

(5.100)

we conclude that we can give an upper bound for the sum in (5.89), that does not depend on $H$.

We have now shown, that for all $(x, y) \in \Omega_{\text{def}}^H$

$$\left| (M_2 g_{\text{hor}}^H)(x, y) \right| \leq (C_1 H^2 + D_1 h^2) \| g_{\text{hor}}^H \|_{\infty},$$

(5.101)
Convergence behavior of the LDC method as an iterative process

with $C_1, D_1$, independent of $H$ and $h$.

The results are summarized in the following lemma, which is the discrete equivalent of Lemma 5.5.

**Lemma 5.10** Consider the LDC algorithm for the Poisson problem (5.14) with the following settings. Let $L^H$ be the standard five-point discretization of the Laplacian on a uniform grid with grid sizes $\Delta x = \Delta y = H$ as in (5.15). Let $\Omega = (0, \gamma)^2$ with $0 < \gamma < 1$ and $\gamma$ a multiple of $H$. Let $L^h$ be the standard five-point discretization of the Laplacian on a uniform grid with grid sizes $\Delta x = \Delta y = h$ as in (5.65). Let $P^{h,H}$ be interpolation by a trigonometric polynomial. Finally, let

$$\Omega^H_{\text{vert}} = \{(x, y) \in \Omega^H | x < \gamma - \epsilon \wedge y < \gamma - \epsilon\},$$

(5.102)

for some $\epsilon > 0$, $\epsilon$ independent of $H$. Then

$$\|M^H g^H\|_{\infty} \leq (C_1 H^2 + D_1 h^2) \|g^H\|_{\infty},$$

(5.103)

for all $g^H \in G(\Gamma^H)$ with $g^H(x, y) \neq 0$ for $(x, y) \in \Gamma^H$ only. In (5.103), $C_1$ and $D_1$ are independent of $H$ and $h$.

Starting from (5.30), we may also assume, that $g_{\text{hor}}^H$ and $g_{\text{corner}}^H$ equal zero. Again, we can prescribe artificial Dirichlet boundary conditions $g_{\text{vert}}^H$ by using interpolation with a trigonometric polynomial. Now, the non-zero boundary conditions are prescribed on the vertical part of the interface. This leads to equivalents of (5.68) and (5.69). An analogue of Lemma 5.10 holds.

**Lemma 5.11** Consider the LDC algorithm for the Poisson problem (5.14) with the following settings. Let $L^H$ be the standard five-point discretization of the Laplacian on a uniform grid with grid sizes $\Delta x = \Delta y = H$ as in (5.15). Let $\Omega = (0, \gamma)^2$ with $0 < \gamma < 1$ and $\gamma$ a multiple of $H$. Let $L^h$ be the standard five-point discretization of the Laplacian on a uniform grid with grid sizes $\Delta x = \Delta y = h$ as in (5.65). Let $P^{h,H}$ be interpolation by a trigonometric polynomial. Finally, let

$$\Omega^H_{\text{vert}} = \{(x, y) \in \Omega^H | x < \gamma - \epsilon \wedge y < \gamma - \epsilon\},$$

(5.104)

for some $\epsilon > 0$, $\epsilon$ independent of $H$. Then

$$\|M^H g^H\|_{\infty} \leq (C_2 H^2 + D_2 h^2) \|g^H\|_{\infty},$$

(5.105)

for all $g^H \in G(\Gamma^H)$ with $g^H(x, y) \neq 0$ for $(x, y) \in \Gamma^H$ only. In (5.105), $C_2$ and $D_2$ are independent of $H$ and $h$.

5.3.2 Treatment of a non-zero value at the corner of the interface

If a vector $g^H \in G(\Gamma^H)$ is such that $g_{\text{corner}}^H$ is not equal to zero, then we can subtract a suitable grid function to accomplish a zero corner value as we did in Section 5.2.3. In fact, the same function as before, cf. (5.57),

$$v(x, y) = \frac{xy}{\gamma^2}, \quad (x, y) \in \Omega_1,$$

(5.106)
5.3 Norm of the iteration matrix—discrete local problem

can be used. We introduce $\tilde{g}_I^H \in G(Γ^H)$ for each vector $g_I^H \in G(Γ^H)$ with $g_I^H(γ, γ) \neq 0$ by (cf. (5.58))

$$\tilde{g}_I^H := g_I^H - g_I^H(γ, γ) \cdot v|_{Γ^H}. \quad (5.107)$$

The grid function $\tilde{g}_I^H$ is zero in the corner $(γ, γ)$ of the interface. Hence, we can split $\tilde{g}_I^H$ according to, cf. (5.30),

$$\tilde{g}_I^H = \tilde{g}_{hor}^h + \tilde{g}_{vert}^h \quad (5.108)$$

and use trigonometric interpolation on both the horizontal and the vertical part of the interface to find artificial Dirichlet boundary conditions $\tilde{g}_{hor}^h \tilde{g}_{vert}^h$, respectively. Similar to Section 5.2.3 we define, cf. (5.60),

$$p_{h,1}^H \tilde{g}_I^H = \tilde{g}_{hor}^h + \tilde{g}_{vert}^h + g_I^H(γ, γ) \cdot v|_{Γ^H}. \quad (5.109)$$

Using this definition, we have

$$M_2 g_I^H = M_2 \tilde{g}_{hor}^H + M_2 \tilde{g}_{vert}^H + g_I^H(γ, γ) \cdot M_2 (v|_{Γ^H}) = M_2 \tilde{g}_{hor}^H + M_2 \tilde{g}_{vert}^H. \quad (5.110)$$

The last equality holds, because the projection of $v$ is discretely harmonic on the local coarse grid $Ω^H$. For the remaining two terms in the right hand side, we have the results from the previous section. This leads to the main theorem.

**Theorem 5.12** Consider the LDC algorithm for the Poisson problem (5.14) with the following settings. Let $L^H$ be the standard five-point discretization of the Laplacian on a uniform grid with grid sizes $Δx = Δy = h$ as in (5.15). Let $Ω^H = (0, γ)^2$ with $0 < γ < 1$ and $γ$ a multiple of $H$. Let $L^H_1$ be the standard five-point discretization of the Laplacian on a uniform grid with grid sizes $Δx = Δy = h$ as in (5.15). Let $p_{h,1}^H$ be trigonometric interpolation. Finally, let

$$Ω^H_{slip} = \{(x, y) \in Ω^H| x < γ - ε \land y < γ - ε\}, \quad (5.111)$$

for some $ε > 0$, $ε$ independent of $H$. Then the following upper bound for the norm of the iteration matrix $M$ of the LDC algorithm holds:

$$\|M\|_\infty \leq CH^2 + Dh^2, \quad (5.112)$$

with $C$, $D$, independent of $H$ and $h$.

**Proof.** Let $g_I^H \in G(Γ^H)$. Define $g_I^H \in G(Γ^H)$ as in (5.107). Using (5.110) and Lemmas 5.10 5.11 we find

$$\|M_2 g_I^H\|_\infty = \|M_2 \tilde{g}_{hor}^H + M_2 \tilde{g}_{vert}^H\|_\infty \leq [(C_1 + C_2)h^2 + (D_1 + D_2)h^2]\|g_I^H\|_\infty,$$

with $C_1$, $C_2$, $D_1$, $D_2$, independent of $H$ and $h$. From the definition of $\tilde{g}_I^H$, it follows that $\|g_I^H\|_\infty \leq 2\|g_I^H\|_\infty$, and hence

$$\|M_2\|_\infty \leq 2(C_1 + C_2)h^2 + 2(D_1 + D_2)h^2.$$

Using (5.19), we conclude that

$$\|M\|_\infty \leq \frac{1}{4}(C_1 + C_2)h^2 + \frac{1}{4}(D_1 + D_2)h^2 = CH^2 + Dh^2,$$

with $C$, $D$, independent of $H$ and $h$. □
5.4 Numerical experiments

In this section, we consider the LDC algorithm described in Section 3.2 with the setting chosen as in Section 5.3 and we verify the theoretical results of Theorem 5.12 in some numerical experiments. For the first experiment, we apply the LDC algorithm to the boundary value problem

\[
\begin{cases}
\Delta u = f, & \text{in } \Omega = (0,1)^2, \\
u = g, & \text{on } \partial \Omega.
\end{cases}
\]

In (5.113), \(f\) and \(g\) have been chosen such that

\[u(x,y) = \tanh[25(x+y-1/3)] + 1.\]

We choose a uniform coarse grid \(\Omega^H\) in \(\Omega\) with grid sizes \(\Delta x = \Delta y = H\) as in (5.15). The coarse grid discretization is the standard five-point scheme. The area of refinement \(\Omega_1\) is chosen as \(\Omega_1 = (0,1/2)^2\). We choose a uniform fine grid \(\Omega^h\) in \(\Omega_1\) with grid sizes \(\Delta x = \Delta y = h\) as in (5.65). The fine grid discretization is the standard five-point scheme. Contrary to the setting in Section 5.3, the operator \(P^h,H\) will be linear interpolation instead of trigonometric interpolation. The reason for this choice is that linear interpolation is more likely to be used in practice. It also shows similar convergence results. The latter will be illustrated in the second experiment.

The results of Theorem 5.12 hold for LDC iteration when the extent \(\epsilon\) of the safety region is larger than zero and independent of \(H\). Hence, we start by choosing \(\epsilon = 1/8\), and we set

\[\Omega^H_{\text{def}} = \{(x,y) \in \Omega^H | x < 1/2 - \epsilon \land y < 1/2 - \epsilon\},\]

see Figure 3.4. We study the convergence rate of the LDC algorithm for \(H = 1/2^k\), \(k = 3,4,\ldots\). The grid size of the local fine grid is chosen as \(h = H/2\). Numerical results are presented in Table 5.1(a). To demonstrate the dependence of the convergence factor on \(H\), we list the maximum differences \(\delta_i\), \(i = 1,2,\ldots\), of the numerical solution at the coarse grid points on the interface, viz.

\[\delta_i := \|u^H_{i;1} - u^H_{i;1-1}\|_{\infty}, \quad i = 1,2,\ldots\]

and the ratios

\[\rho_i := \frac{\delta_i}{\delta_{i-1}}, \quad i = 2,3,\ldots\]

The motivation for considering the ratios \(\rho_i\) is the following. For a scalar linearly convergent process, say

\[x_i \to \alpha, \quad (i \to \infty),\]

we have

\[\frac{|x_i - \alpha|}{|x_{i-1} - \alpha|} \to C, \quad (i \to \infty),\]

with \(C\) constant. The constant \(C\) is called the convergence factor, and we have \(C \in [0,1)\). It is easily verified that

\[\frac{|x_i - x_{i-1}|}{|x_{i-1} - x_{i-2}|} \to C, \quad (i \to \infty).\]
5.4 Numerical experiments

(a) Maximum differences on the interface.

<table>
<thead>
<tr>
<th>H</th>
<th>ρ1</th>
<th>ρ2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/8</td>
<td>8.66 · 10^{-1}</td>
<td>2.08 · 10^{-3}</td>
</tr>
<tr>
<td>1/16</td>
<td>1.33 · 10^{-1}</td>
<td>1.20 · 10^{-4}</td>
</tr>
<tr>
<td>1/32</td>
<td>2.70 · 10^{-3}</td>
<td>7.55 · 10^{-7}</td>
</tr>
<tr>
<td>1/64</td>
<td>3.39 · 10^{-4}</td>
<td>2.53 · 10^{-8}</td>
</tr>
<tr>
<td>1/128</td>
<td>7.16 · 10^{-5}</td>
<td>1.40 · 10^{-9}</td>
</tr>
<tr>
<td>1/256</td>
<td>1.63 · 10^{-5}</td>
<td>8.13 · 10^{-11}</td>
</tr>
<tr>
<td>1/512</td>
<td>3.86 · 10^{-6}</td>
<td>4.87 · 10^{-12}</td>
</tr>
</tbody>
</table>

(b) Estimates of the convergence factor.

<table>
<thead>
<tr>
<th>H</th>
<th>ρ2</th>
<th>ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/8</td>
<td>2.41 · 10^{-3}</td>
<td></td>
</tr>
<tr>
<td>1/16</td>
<td>9.03 · 10^{-4}</td>
<td>0.38</td>
</tr>
<tr>
<td>1/32</td>
<td>2.79 · 10^{-4}</td>
<td>0.31</td>
</tr>
<tr>
<td>1/64</td>
<td>7.47 · 10^{-5}</td>
<td>0.27</td>
</tr>
<tr>
<td>1/128</td>
<td>1.95 · 10^{-5}</td>
<td>0.26</td>
</tr>
<tr>
<td>1/256</td>
<td>5.00 · 10^{-6}</td>
<td>0.26</td>
</tr>
<tr>
<td>1/512</td>
<td>1.26 · 10^{-6}</td>
<td>0.25</td>
</tr>
</tbody>
</table>

Table 5.1: Estimates of the convergence factor for an H-independent safety zone (ε = 1/8, h = H/2, linear interpolation).

The limit (5.120) provides a way to estimate the convergence factor C. In our experiment, we will see that the vectors \( u^H_{i+1} \) converge to \( u^H_i \) linearly. The ratios

\[
\rho_i = \frac{\|u^H_{i+1} - u^H_{i+1-1}\|_\infty}{\|u^H_{i+1-1} - u^H_{i+1-2}\|_\infty}
\]

(5.121)

are the generalization of the ratios in (5.120). Hence, they provide a means to estimate the convergence factor.

In our experiment, the ratios \( \rho_i \) are nearly constant for fixed H, when they are not effected by rounding errors. In Table 5.1(b), we show \( \rho_2 \) and their quotients for different grid sizes H. The ratios indicate that the convergence factor is \( O(H^2) \), as predicted by Theorem 5.12, indeed.

One of the main assumptions of Theorem 5.12 is that the extent of the safety zone ε is positive and independent of H. Repeating the experiment with ε = 0, i.e., with no safety zone, shows that this assumption is necessary indeed. Table 5.2(a) lists estimates of the convergence factor for ε = 0. Although the ratios \( \rho_1 \) are still small, implying fast convergence, they are not \( O(H^2) \). The results seem to indicate that the convergence factor is \( O(H) \) rather than \( O(H^2) \). Similar results are found when we take ε positive, but not independent of H; Table 5.2(b) lists estimates of the convergence factor for ε = 2H.

In all three Tables 5.1(b), 5.2(a,b), we used h = H/2. If we only change H and keep
Convergence behavior of the LDC method as an iterative process

(a) No safety zone \( (\epsilon = 0, h = H/2) \).

(b) Safety zone dependent on \( H \) \((\epsilon = 2H, h = H/2) \).

(c) Constant \( h \) \((\epsilon = 1/8, h = 1/1024) \).

<table>
<thead>
<tr>
<th>( H )</th>
<th>( p_2 ) ratio</th>
<th>( H )</th>
<th>( p_2 ) ratio</th>
<th>( H )</th>
<th>( p_2 ) ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/8</td>
<td>1.12 \cdot 10^{-2}</td>
<td>1/8</td>
<td>1.18 \cdot 10^{-4}</td>
<td>1/8</td>
<td>2.91 \cdot 10^{-3}</td>
</tr>
<tr>
<td>1/16</td>
<td>1.20 \cdot 10^{-2}</td>
<td>1/16</td>
<td>9.03 \cdot 10^{-4}</td>
<td>1/16</td>
<td>1.20 \cdot 10^{-3}</td>
</tr>
<tr>
<td>1/32</td>
<td>3.56 \cdot 10^{-3}</td>
<td>1/32</td>
<td>4.31 \cdot 10^{-4}</td>
<td>1/32</td>
<td>3.72 \cdot 10^{-3}</td>
</tr>
<tr>
<td>1/64</td>
<td>8.79 \cdot 10^{-3}</td>
<td>1/64</td>
<td>1.02 \cdot 10^{-4}</td>
<td>1/64</td>
<td>9.91 \cdot 10^{-5}</td>
</tr>
<tr>
<td>1/128</td>
<td>3.56 \cdot 10^{-3}</td>
<td>1/128</td>
<td>2.47 \cdot 10^{-4}</td>
<td>1/128</td>
<td>2.56 \cdot 10^{-5}</td>
</tr>
<tr>
<td>1/512</td>
<td>8.97 \cdot 10^{-4}</td>
<td>1/512</td>
<td>3.39 \cdot 10^{-5}</td>
<td>1/512</td>
<td>6.25 \cdot 10^{-6}</td>
</tr>
</tbody>
</table>

Table 5.2: Estimates of the convergence factor for different values of \( \epsilon \) and \( h \) (linear interpolation in all tables).

h fixed on a small value, we find almost identical results. This is demonstrated in Table 5.2(c), where we have chosen \( \epsilon = 1/8 \) again and where we have fixed \( h = 1/1024 \). The estimates of the convergence factor are almost the same as in Table 5.1(b).

Similar numerical experiments have been presented by Hackbusch [Hackbusch, 1984, Sec. 3.2.1] and Wappler [Wappler, 1999, Sec. 3.3.4.1]. Both authors also find the convergence factor to be \( O(H^2) \) with \( H \) the size of the coarse grid, provided the extent of the safety region \( \epsilon \) is larger than zero and independent of \( H \). Wappler also studies the dependence of the convergence factor on \( \epsilon \) when \( \epsilon \) depends on \( H \) linearly. His results do not show a clear convergence behavior. Wappler conjectures that the convergence factor tends to a constant unequal to either zero or unity for \( H \) tending to zero.

Rather than applying the LDC algorithm to a specific problem and studying the convergence factor for this particular situation, it is also possible to directly construct the iteration matrix \( M \) of Theorem 5.1 and calculate its norm. For the setting chosen as above, in particular linear interpolation, \( \epsilon = 1/8 \), and \( h = H/2 \), we find the numerical results as in Table 5.3. Listed are the infinity norms of the iteration matrix \( M \) from Theorem 5.1 and of the matrices \( M_1 \) and \( M_2 \) as defined by (5.18). We observe that

\[
\|M\|_\infty = O(H^2),
\]

(5.122)

which is in perfect agreement with Theorem 5.12. Note that we have used the upper bound 1/8 for \( \|M_1\|_\infty \) cf. (5.19), which is not very sharp. As we are interested in the asymptotic behavior for \( H \) tending to zero, however, the matrix \( M_1 \) is not very interesting, because its norm is almost constant for decreasing \( H \). The results of Sections 5.2 and 5.3 were targeted at estimating the norm of \( M_2 \), which appears to be \( O(H^2) \) indeed.

If we consider the LDC algorithm with no safety region, i.e., \( \epsilon = 0 \), we find the results of Table 5.4. Table 5.4 shows that the norm of the iteration matrix increases for decreasing \( H, h \). Hence, the convergence factor deteriorates for smaller values of \( H \) and \( h \). Note that the convergence factor for the LDC algorithm with no safety region applied to boundary value problem (5.113) is much better than one would expect from Table 5.4 cf. Table 5.2(a).
### 5.4 Numerical experiments

The estimates of the convergence factor \( \rho_i \) in the first experiments and the iteration matrix \( M \) in the second experiments are related in the following way:

\[
\rho_i = \frac{\|e_{i+1} - e_{i+1}^H\|_{\infty}}{\|e_{i} - e_{i}^H\|_{\infty}} \leq \|M\|_{\infty}.
\]  
(5.123)

This upper bound can be verified by comparing Table 5.1 with Table 5.3 and by comparing Table 5.2(a) with Table 5.4 respectively. Indeed, all \( \rho_i \) are less than the norms \( \|M\|_{\infty} \). The bound is not sharp.

We finally present the matrix norms for the standard experiment with \( \epsilon = 1/8 \) and \( h = H/2 \), when we use interpolation by trigonometric polynomials instead of linear interpolation. Comparing Table 5.3 (results for linear interpolation) and Table 5.5 (results for trigonometric interpolation) confirms the claim made at the beginning of this section: using linear interpolation for \( \Phi_h^H \) gives similar results as using trigonometric interpolation.

<table>
<thead>
<tr>
<th>( h )</th>
<th>( |M|_{\infty} )</th>
<th>ratio</th>
<th>( |M_1|_{\infty} )</th>
<th>ratio</th>
<th>( |M_2|_{\infty} )</th>
<th>ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/8</td>
<td>8.30 \times 10^{-3}</td>
<td></td>
<td>1.29 \times 10^{-2}</td>
<td></td>
<td>6.66 \times 10^{-1}</td>
<td></td>
</tr>
<tr>
<td>1/16</td>
<td>4.07 \times 10^{-5}</td>
<td>0.49</td>
<td>1.78 \times 10^{-2}</td>
<td>1.38</td>
<td>4.98 \times 10^{-1}</td>
<td>0.75</td>
</tr>
<tr>
<td>1/32</td>
<td>1.36 \times 10^{-3}</td>
<td>0.33</td>
<td>2.04 \times 10^{-2}</td>
<td>1.14</td>
<td>2.65 \times 10^{-1}</td>
<td>0.53</td>
</tr>
<tr>
<td>1/64</td>
<td>4.04 \times 10^{-4}</td>
<td>0.30</td>
<td>2.17 \times 10^{-2}</td>
<td>1.06</td>
<td>9.49 \times 10^{-1}</td>
<td>0.36</td>
</tr>
<tr>
<td>1/128</td>
<td>1.11 \times 10^{-4}</td>
<td>0.27</td>
<td>2.23 \times 10^{-2}</td>
<td>1.03</td>
<td>2.97 \times 10^{-1}</td>
<td>0.31</td>
</tr>
</tbody>
</table>

Table 5.3: Matrix norms for LDC iteration with an \( H \)-independent safety zone (\( \epsilon = 1/8 \), \( h = H/2 \), linear interpolation).

<table>
<thead>
<tr>
<th>( h )</th>
<th>( |M|_{\infty} )</th>
<th>ratio</th>
<th>( |M_1|_{\infty} )</th>
<th>ratio</th>
<th>( |M_2|_{\infty} )</th>
<th>ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/8</td>
<td>1.64 \times 10^{-1}</td>
<td></td>
<td>1.29 \times 10^{-2}</td>
<td></td>
<td>6.93 \times 10^{-1}</td>
<td></td>
</tr>
<tr>
<td>1/16</td>
<td>2.38 \times 10^{-1}</td>
<td>1.45</td>
<td>1.78 \times 10^{-2}</td>
<td>1.38</td>
<td>2.80 \times 10^{-1}</td>
<td>4.03</td>
</tr>
<tr>
<td>1/32</td>
<td>3.05 \times 10^{-1}</td>
<td>1.28</td>
<td>2.04 \times 10^{-2}</td>
<td>1.14</td>
<td>1.12 \times 10^{-1}</td>
<td>4.00</td>
</tr>
<tr>
<td>1/64</td>
<td>3.70 \times 10^{-1}</td>
<td>1.21</td>
<td>2.17 \times 10^{-2}</td>
<td>1.06</td>
<td>4.48 \times 10^{-1}</td>
<td>4.00</td>
</tr>
<tr>
<td>1/128</td>
<td>4.34 \times 10^{-1}</td>
<td>1.17</td>
<td>2.23 \times 10^{-2}</td>
<td>1.03</td>
<td>1.79 \times 10^{-1}</td>
<td>4.00</td>
</tr>
</tbody>
</table>

Table 5.4: Matrix norms for LDC with no safety zone (\( \epsilon = 0 \), \( h = H/2 \), linear interpolation).

<table>
<thead>
<tr>
<th>( h )</th>
<th>( |M|_{\infty} )</th>
<th>ratio</th>
<th>( |M_1|_{\infty} )</th>
<th>ratio</th>
<th>( |M_2|_{\infty} )</th>
<th>ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/8</td>
<td>8.28 \times 10^{-3}</td>
<td></td>
<td>1.29 \times 10^{-2}</td>
<td></td>
<td>6.35 \times 10^{-1}</td>
<td>0.69</td>
</tr>
<tr>
<td>1/16</td>
<td>3.82 \times 10^{-3}</td>
<td>0.46</td>
<td>1.78 \times 10^{-2}</td>
<td>1.38</td>
<td>4.35 \times 10^{-1}</td>
<td>0.58</td>
</tr>
<tr>
<td>1/32</td>
<td>1.33 \times 10^{-3}</td>
<td>0.35</td>
<td>2.04 \times 10^{-2}</td>
<td>1.14</td>
<td>2.52 \times 10^{-1}</td>
<td>0.37</td>
</tr>
<tr>
<td>1/64</td>
<td>4.01 \times 10^{-4}</td>
<td>0.30</td>
<td>2.17 \times 10^{-2}</td>
<td>1.06</td>
<td>9.53 \times 10^{-1}</td>
<td>0.37</td>
</tr>
<tr>
<td>1/128</td>
<td>1.11 \times 10^{-4}</td>
<td>0.28</td>
<td>2.23 \times 10^{-2}</td>
<td>1.03</td>
<td>2.96 \times 10^{-1}</td>
<td>0.32</td>
</tr>
</tbody>
</table>

Table 5.5: Matrix norms for LDC with an \( H \)-independent safety zone and trigonometric interpolation (\( \epsilon = 1/8 \), \( h = H/2 \), trigonometric interpolation).
Chapter 6

Adaptive multi-level LDC in combination with domain decomposition

In this chapter, we present a technique to discretize and solve elliptic boundary value problems on composite grids found by adaptive grid refinement. In this technique, we will only use standard discretizations on rectangular tensor-product grids. The full algorithm is obtained by successively adding adaptivity, multi-level refinement, domain decomposition and regridding to the standard LDC algorithm as presented in Chapter 3.

The algorithm is started by calculating an initial solution on a coarse tensor-product grid. We consider the cells formed by grid points of this coarse grid, and determine which cells lie in a high activity area of the solution. These high activity boxes are flagged for refinement based on a weight function that is an indicator for the smoothness of the solution. To prevent the solution from being artificially trapped at interfaces between coarse and fine grids, we also label neighboring boxes of high activity boxes for refinement. The flagged boxes are covered with a rectangular patch. In the patch, a finer grid is chosen. The initial coarse grid solution is interpolated to find both an initial guess on the fine grid and to find artificial Dirichlet boundary conditions on the interface between the fine and the coarse grids. The boundary value problem is discretized and solved on the fine grid in the patch.

When the problem has been solved on the finer grid, the more accurate solution found locally is used to estimate the local discretization error of the coarse grid discretization in the corners of the flagged boxes of the coarse grid. Next, the approximation on the initial coarse tensor-product grid is updated using this estimate for the discretization error. Having found a new approximation on the initial tensor-product grid, we may again proceed to solve the boundary value problem on the finer local grid, now using the
updated coarse grid solution to find more accurate boundary conditions on the interface between the fine and the coarse grids.

This gives rise to an iterative solution method: the new fine grid approximation can be used to improve the coarse grid approximation, and the new coarse grid approximation provides better boundary conditions on the interface, and hence a more accurate fine grid approximation. The solution procedure may be applied recursively, i.e., the rectangular patches used to cover high activity areas in the coarse grid may be refined themselves. The maximum level of refinement can be chosen such that the desired accuracy on the composite grid is achieved.

The usage of a single rectangular patch to cover all flagged boxes in a grid may be inefficient. The reason is that it may cause refinement of unflagged boxes. To remedy this inefficiency as well as to prevent the grids from becoming too large, we combine the multi-level LDC algorithm with domain decomposition.

Finally, we note that refining a grid and solving the boundary value problem on the new composite grid may cause the area of high activity to move. When this happens, areas of the grid may be refined unnecessarily or areas may not be refined whereas they do require refinement. Therefore, we formulate a regridding procedure.

### 6.1 Adaptive multi-level refinement

In the standard LDC algorithm presented in Section 3.2, see Algorithm 3.1, we assumed that the area of refinement was a priori chosen. We will now determine the area of refinement as part of the algorithm. To formulate the extra step in the algorithm, we consider the model problem from Section 3.2 again, cf. (3.10),

\[
\begin{aligned}
Lu &= f, & \text{in } \Omega, \\
u &= g, & \text{on } \partial \Omega.
\end{aligned}
\]  

(6.1)

We present an algorithm for discretizing and solving (6.1) on a composite grid found by adaptive grid refinement under the assumption that we have a code for solving boundary value problem (6.1) on a tensor-product grid in a rectangular domain. For ease of presentation, we choose the domain \( \Omega \) 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 set \( x_i := IH, y_j := jH, i,j = 0, 1, 2, \ldots, N \), where the grid size \( H \) is chosen as \( H := 1/N \), with \( N \) a positive integer. The initial global coarse grid is denoted by \( \Omega^H \) and is defined by

\[
\Omega^H := \{(x_i, y_j) \mid i, j = 1, 2, \ldots, N-1\}.
\]  

(6.2)

An initial approximation \( u_{0}^{H} \) on \( \Omega^{H} \) can be found by solving the system (cf. (3.11))

\[
L^H u_{0}^{H} = f^{H},
\]  

(6.3)

which is a discretization of boundary value problem (6.1).
We assume that the continuous solution \( u \) of (6.1) has a high activity region in some (small) part of the domain. This high activity of \( u \) may be captured by discretizing (6.1) on a composite grid. Following Bennett and Smooke [Bennett and Smooke, 1998, Bennett and Smooke, 1999, Valdati, 1997], we introduce a positive weight function \( w \) as an indicator function for the smoothness of the solution, and we base the adaptive gridding on subequidistribution of this weight function. Additional grid points will be added in regions where \( w \) is large; \( w \) should therefore measure the rapidity of change of \( u \).

For one-dimensional problems, Kautsky and Nichols [Kautsky and Nichols, 1980] formulate the objective of adaptive grid procedures with equidistribution techniques as follows. Given a positive weight function defined on an interval, find a partitioning of the interval such that the integral of the weight function takes a given constant value over each subinterval. The authors develop a technique for creating a grid that satisfies certain smoothness constraints. One may think of quasi-uniformity, meaning that the ratio of the maximal to the minimal step in the grid is bounded, and local boundedness, meaning that the ratio of adjacent steps is bounded.

The boxes \( B_{ij} \) are defined as the cells formed by grid points and points on the boundary of the domain, viz.

\[
B_{ij} = (x_i, x_{i+1}) \times (y_j, y_{j+1}), \quad i, j = 0, 1, \ldots, N - 1. \tag{6.4}
\]

We choose the weight function such that it assigns a positive number measuring the smoothness of \( u \) in each box of the grid. Following Bennett and Smooke, we require the value of the weight function in a box to depend on the values of the dependent variable \( u \) at the four corners of the box only. Hence, it includes derivatives up to first order. Inclusion of higher order derivatives in the weight function may give a better estimator of the local error, but would need a larger stencil.

The (first) derivatives used may contain some numerical fluctuation; to remedy this fluctuation, we will apply a smoothing filter in what follows. We start by defining the grid function \( w_{ij}^u \). The superscript ‘\( u \)’ stands for ‘unsmeothed’. In its definition, we will use the (averaged) finite difference approximations

\[
\frac{\delta u}{\delta x} \bigg|_{B_{ij}} = \frac{1}{2} \left( \frac{u_{i+1,j} - u_{i,j}}{H} + \frac{u_{i+1,j+1} - u_{i,j+1}}{H} \right), \tag{6.5}
\]

\[
\frac{\delta u}{\delta y} \bigg|_{B_{ij}} = \frac{1}{2} \left( \frac{u_{i,j+1} - u_{i,j}}{H} + \frac{u_{i+1,j+1} - u_{i+1,j}}{H} \right). \tag{6.6}
\]

The value of \( w_{ij}^u \) in box \( B_{ij} \) is defined by

\[
w_{ij}^u := 1 + \alpha \max_k \sqrt{\left( \frac{\delta u}{\delta x} \bigg|_{B_{ik}} \right)^2 + \left( \frac{\delta u}{\delta y} \bigg|_{B_{ij}} \right)^2}, \tag{6.7}
\]

in which \( \alpha \) is a user-specified coefficient. According to [Bennett and Smooke, 1998, Bennett and Smooke, 1999], the constant 1 in (6.7) tempers solution adaptivity with
Adaptive multi-level LDC in combination with domain decomposition

grid uniformity. In practice, we choose $\alpha = 10^6$. A subsequent normalization process (cf. (6.10)) makes the adaptive grids largely insensitive to the specific value of $\alpha$, as long as $\alpha \gg 1$. The maximum in the denominator of the second term at the right hand side of (6.7) is over all boxes $B_{kl}$, $k, l = 0, 1, \ldots, N-1$. If the dependent variable $u$ is constant over the domain, $w^u$ is defined as unity for all boxes.

We apply a smoothing filter to the weight function values; the filter is found by discretization of the Laplace equation. The equation $\Delta w = 0$ is discretized by central differences, and the resulting equation is solved for the value in the central box. On a uniform grid and for interior points $(i, j) = 1, 2, \ldots, N-2$, this filter is defined through

$$w_{ij}^t := \frac{w_{i+1,j}^u + w_{i-1,j}^u + w_{i,j+1}^u + w_{i,j-1}^u}{4},$$

(6.8)

in which the superscript ‘$i$’ stands for ‘intermediate’. The smoothed weight function $w_s^i$ is found by averaging the unsmoothed value $w_{ij}^u$ with the intermediate value $w_{ij}^t$, viz.

$$w_s^i := \frac{w_{ij}^u + w_{ij}^t}{2},$$

(6.9)

Application of the smoothing filter (6.8) followed by the averaging procedure (6.9) can also be viewed as applying one step of damped Jacobi iteration. Damped Jacobi iteration is known to be a smoother in multigrid literature. At the boundaries (boxes $B_{ij}$ for which exactly one of the conditions $i = 0, i = N-1, j = 0, \text{or } j = N-1$ is true), the weight function is smoothed in one direction using information along the boundary only. In the four corner boxes, the weight function remains unchanged. The smoothing filter (6.8), (6.9) may be applied to the values $w_{ij}^t$ again; the number of smoothing passes must be specified and is typically equal to 10.

The goal of the adaptive gridding is to create a grid on which the weight function is subequidistributed. If we normalize the weight function values by

$$w_{ij} := \frac{w_{ij}^s}{\bar{w}}, \quad \bar{w} := \frac{1}{N^2} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} w_{kl}^s,$$

(6.10)

equidistribution means that $w_{ij} = 1$ for all $i, j = 0, 1, \ldots, N-1$. We define high activity boxes as those boxes for which the normalized weight function exceeds some threshold value; we flag a box $B_{ij}$ for refinement if $w_{ij} > \epsilon$, where the threshold $\epsilon$ is a user-specified parameter typically ranging from 1.5 to 3.

Due to the refinement of the grid, the solution can move, causing high activity areas to be located at different positions. This phenomenon occurs e.g. in the application problem of the axisymmetric Bunsen flame, as we will see in Chapter 7. To prevent the solution from being artificially trapped at interfaces between coarse and fine grids, we also label neighboring boxes of high activity boxes for refinement. The number of neighboring boxes of a high activity box that are being flagged in each direction is a parameter of the algorithm, which we denote by $N_{\text{neighbor}}$. We find the following
6.1 Adaptive multi-level refinement

collection of all flagged boxes:

\[ B_{\text{flagged}} = \{ B_{ij} \mid i, j = 0, 1, \ldots, N - 1, \]
\[ \exists k, l : k, l = 0, 1, \ldots, N - 1, \]
\[ i - N_{\text{neighbor}} \leq k \leq i + N_{\text{neighbor}}, \]
\[ j - N_{\text{neighbor}} \leq l \leq j + N_{\text{neighbor}} : w_{kl} > \epsilon \} \].

The idea of flagging neighboring boxes is called layering in Bennett and Smooke, 1998, Bennett and Smooke, 1999. Note that the thickness of the layer around high activity boxes is expressed in fine grid boxes in Bennett and Smooke, 1998, Bennett and Smooke, 1999 rather than in coarse grid boxes as we do in (6.11). Bennett and Smooke choose to surround the original high activity boxes by a number of \( N_{\text{layer}} \) fine grid boxes. We will elaborate on this when we formulate the solution procedure.

Ideally, the value of the refinement factor \( \sigma \) should be chosen largest at places where the weight function is largest. Because our goal is equidistribution of the weight function on the composite grid, boxes with large values of the weight function should be divided in many smaller boxes. This would lead to an unstructured composite grid though. To avoid the usage of an unstructured grid, we will not vary \( \sigma \) per box. Instead, we will use more than one level of refinement. In this approach, the finer grid may be refined itself. This is considered in detail in what follows.

We assume that the continuous solution \( u \) has one area of high activity. It is straightforward to generalize the algorithm to deal with the case where there is more than one area of high activity. We cover the flagged boxes by the smallest enclosing rectangle. Hence, we set

\[ \Omega_1 = (x_{\text{min}}, x_{\text{max}}) \times (y_{\text{min}}, y_{\text{max}}), \]

where \( x_{\text{min}}, x_{\text{max}} \) are defined by

\[ x_{\text{min}} = \min_{B_{ij} \in B_{\text{flagged}}} x_{ij}, \quad x_{\text{max}} = \max_{B_{ij} \in B_{\text{flagged}}} x_{ij + 1}, \]

and \( y_{\text{min}}, y_{\text{max}} \) are defined in a similar way. In \( \Omega_1 \), we choose a local fine grid by uniform refinement of all boxes that constitute \( \Omega_1 \) with an integer refinement factor denoted by \( \sigma \). Typically, \( \sigma \) equals 2, although values of \( \sigma \) larger than 2 are allowed. For the setting chosen in this section, the local fine grid has grid size \( h = H/\sigma \), and we will denote the grid by \( \Omega_1^h \) as in Section 3.2.

In the LDC Algorithm 3.1, see Section 3.2, the local fine grid is a priori chosen. The corresponding iterative method with adaptive gridding can be formulated as follows.

**Algorithm 6.1**

**Two-grid LDC algorithm with adaptive gridding**

**Initialization**

- Solve the basic coarse grid problem (3.11).
Adaptive multi-level LDC in combination with domain decomposition

- Determine the location of the high activity area in the coarse grid and choose a fine grid in this area.
- Solve the local fine grid problem (3.12) with $i = 0$.

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

- Solve the updated coarse grid problem (3.20).
- Solve the local fine grid problem (3.12).

We now generalize the algorithm to the situation that we use more than one level of refinement: the finer grid covering the area of high activity of the base grid may itself be refined. 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. The Level $1$ grid in turn may be refined using the same procedure recursively to find a yet finer Level $2$ grid. In this way, we can keep adding levels of refinement until the finest grid has the resolution that is desired.

It should be noted that the procedure does not need to be identical at all levels. Indeed, one can choose the parameters $\epsilon$ and $N_{\text{neighbor}}$ to be dependent on the level. In practice, we keep $\epsilon$ fixed at all levels at the value $\epsilon = 2$; $N_{\text{neighbor}}$ will depend on the level.

The layering procedure was introduced to provide a buffering layer so that high activity areas of the solution can move to a different location. To make sure that the size of the layer is sufficiently large at all levels, the layering parameter should be dependent on the level $l$, see Bennett and Smooke [Bennett and Smooke, 1999]. They choose to surround the originally flagged high activity area with $N_{\text{layer}}$ 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 [Bennett and Smooke, 1999, form. (6)]:

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

(6.14)

Note that the minimum in both [Bennett and Smooke, 1999, form. (6)] and [Bennett and Smooke, 1999, form. (7)] is a misprint; it should be replaced with a maximum. This particular choice (6.14) for the layering parameter is motivated in [Bennett and Smooke, 1999]. If the layering parameter on the Level $0$ grid is chosen larger than $2$, the series $N_{\text{layer}}(l)$, $l = 0, 1, 2, \ldots$, is increasing. Because we have expressed the thickness of the layer in coarse grid boxes, see (6.11), we define

$$N_{\text{neighbor}}(l) := \frac{N_{\text{layer}}(l)}{\sigma}, \quad l = 0, 1, 2, \ldots$$

(6.15)

in which ‘div’ stands for integer division. We propose the more general expressions

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

(6.16)

$$N_{\text{neighbor}}(l) := \frac{N_{\text{layer}}(l)}{\sigma}, \quad l = 0, 1, 2, \ldots$$

(6.17)

because LDC allows the use of refinement factors larger than $2$.

We start the solution procedure by the initialization on the grid at Level $0$. Next, we add levels of refinement, and proceed from Level $0$ to Level $1$, from Level $1$ to Level $2$, etc.
Figure 6.1: Solution procedure for a base grid with two additional levels of refinement.

etc. until we reach the maximum level. Here, we will assume that the maximum level of refinement, denoted by \( l_{\text{max}} \), is a priori chosen. In practice, we run several simulations, successively adding levels of refinement, in order to choose an appropriate value.

After adding successive levels of refinement, we update the solution on the coarser grids: the approximations found on the finer grids are used to improve the approximations on the coarser grids by coarse grid correction steps. Once we have returned to the base grid, we will solve discrete problems on finer levels again. As we have already chosen the location of the subgrid when proceeding to level 1 the first time, we use the same subgrid. This approach is only valid if the area of interest has not changed. If the area has changed, new locations should be chosen; this is detailed in Section 6.5.

At intermediate levels (levels that are not the coarsest nor the finest level), one LDC iteration step involves two solves (see Figure 6.1): 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 first of these solves (the correction solve), defects are calculated. These defects are reused in the second solve. The full solution process is shown in Figure 6.1 for the situation that \( l_{\text{max}} = 2 \). The multi-level LDC algorithm can be formulated as follows.

**Algorithm 6.2**

**Multi-level LDC algorithm**

**Initialization**

- Solve the basic coarse grid problem (3.11) (the Level 0 grid problem).
- For \( l = 0, 1, \ldots, l_{\text{max}} - 1 \), do
  - Determine the location of the high activity area in the Level \( l \) grid and choose the finer Level \( l + 1 \) grid in this area.
  - Solve the Level \( l + 1 \) grid problem.
Adaptive multi-level LDC in combination with domain decomposition

6.2 An example problem

As an illustration of the solution procedure described in the previous sections, we apply the multi-level LDC algorithm to a simple model problem. Consider the boundary value problem

\[ \begin{align*}
    \Delta u &= f, & \text{in } \Omega = (0,1)^2, \\
    u &= g, & \text{on } \partial \Omega.
\end{align*} \]  

(6.18)

In (6.18), \( f \) and \( g \) have been chosen such that

\[ u(x,y) = \exp(\alpha(x-1)) \cdot \exp(-\beta(y-1/2)^2), \quad \alpha = \beta = 50. \]  

(6.19)

The continuous solution \( u \) of the boundary value problem has a sharp peak at \((x,y) = (1,1/2)\), and is smooth in the rest of the domain, see Figure 6.2. We choose a uniform coarse grid \( \Omega^1 \) in \( \Omega \) with grid sizes \( \Delta x = \Delta y = \Delta = 1/10 \). We carry out five different experiments; the difference between these runs is the maximum level of refinement in the LDC algorithm. In all experiments, the threshold value \( \epsilon \) for the weight function is set to 2 at all levels and the layering parameter is chosen according to (6.14), (6.15) with \( N_{\text{layer}}(0) \) equal to 4. The refinement factor \( \sigma \) equals 2. Note that this means that \( N_{\text{neighbor}} \)
6.2 An example problem

![Flagged boxes](image1.png)

(a) Flagged boxes.

![Corresponding LDC grid](image2.png)

(b) Corresponding LDC grid.

Figure 6.3: In the left figure, black boxes are high activity boxes, gray boxes are neighbors and white boxes are unflagged boxes. The edges of the smallest enclosing rectangle are drawn with a thick line. The resulting LDC grid is shown in the right figure.

equals 2, 3, and 5, at Level 0, 1, and 2, respectively. This is illustrated in Figure 6.3(a); it shows the process of flagging boxes for these settings. On all grids, boundary value problem (6.18) is discretized using the standard five-point scheme. Figure 6.3(b) shows the LDC composite grid and the error of the LDC solution for the experiment with the maximum level of refinement $l_{\text{max}}$ equal to 3.

For comparison reasons, approximations have been calculated on uniform grids with grid sizes equal to the grid sizes of the finest grids of the LDC composite grids. All numerical results are listed in Table 6.1. The errors listed are maximum errors and the number of LDC iterations is set to 1. We find that the error on the LDC composite grids is practically equal to the error on the equivalent uniform grid. It is clear that the LDC grids use far less grid points.

The entries in the fourth columns of Tables 6.1(a) and 6.1(b) are the execution times of the calculations in seconds. We listed the average of the solution times of ten identical runs for each experiment in order to eliminate fluctuations due to other processes running on the same computer. For the practical implementation of the LDC algorithm, 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. For this reason, we see that the execution times of the LDC algorithm are larger than those of the uniform grid solver on small grids. LDC performs better than the uniform grid solver with respect to execution time in the last two experiments.
Adaptive multi-level LDC in combination with domain decomposition

Table 6.1: Maximum norms of the error and execution times of the numerical experiments for problem (6.18).

6.3 Domain decomposition

In the previous section, we outlined how to calculate the weight function on a grid, flag boxes for refinement and determine the smallest rectangle enclosing all flagged boxes, cf. (6.11), (6.12). Also, we chose to refine this smallest enclosing rectangle entirely. Note however that this implies that we may be refining many boxes that have not been flagged for refinement, especially when an area of high activity is not aligned with the grid directions. For this reason, refining the entire enclosing rectangle may be quite inefficient. To remedy this inefficiency as well as to prevent the grids from becoming too large, we propose to combine the multi-level LDC algorithm with domain decomposition.

Rather than refining the smallest enclosing rectangle of all flagged boxes in a grid, we use a set of rectangles to cover all boxes that have been flagged for refinement. We denote the set of (overlapping) rectangles by \( S \). It is clear that for each flagged box \( B_{ij} \in \text{flagged} \), there should be at least one rectangle \( R \in S \), such that \( B_{ij} \subset R \). Moreover, we want the rectangles in \( S \) to be overlapping. The overlap of the rectangles is necessary in situations where a high activity region is cut into two or more rectangles. Interfaces between rectangles may intersect high activity zones. Large errors at these interfaces can be remedied by performing a number of domain decomposition iterations. We will use a standard Schwarz alternating procedure. The domain decomposition algorithm we will employ is known as a multiplicative Schwarz procedure. In methods of this type, one alternates between subdomains, solving a new problem each time with boundary conditions updated from the most recent subdomain solutions, see e.g. [Saad, 1996, Ch. 13].

To decide whether we should refine all boxes in the smallest enclosing rectangle of the flagged boxes or whether we should use more than one rectangle at one level, we use the following domain decomposition algorithm. Say we would like to cover all flagged boxes in some coarse grid \( G \). First, we determine the smallest enclosing rectangle covering all flagged boxes. We denote this rectangle by \( R \). The cost we associate with \( R \) is the number of nodes of the finer grid found by dividing all coarse grid boxes contained in \( R \) with the refinement factor \( \sigma \). As an alternative to using the whole rectangle \( R \), we consider splitting \( R \) into two smaller rectangles. If \( R \) can be written as

\[
R = (x_{i_1}, x_{i_2}) \times (y_{j_1}, y_{j_2}),
\]

(6.20)
we could decide to split \( R \) vertically along the index \( i \), \( i_1 < i < i_2 \), into

\[
R_1(i) = (x_{i_1}, x_i) \times (y_{j_1}, y_{j_2}), \quad R_2(i) = (x_i, x_{i_2}) \times (y_{j_1}, y_{j_2}). \quad (6.21)
\]

The cost we associate with this vertical splitting along index \( i \) is based on the costs of the individual grids in \( R_1(i) \) and \( R_2(i) \). These individual costs are defined as follows.

The cost of \( R_1(i) \), notation \( c(R_1(i)) \), is the number of nodes of the finer grid found by refining the smallest rectangle that encloses all flagged coarse grid boxes that lie 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 overlap of subgrids. We define the cost of \( R_2(i) \) in a similar way; for \( R_2(i) \), the overlap is added to the left side of the rectangle though. The cost of vertical splitting can be defined in various ways. A natural choice would be to set

\[
c_{\text{vert}}^{(1)} := \min_{i_1 < i < i_2} \left( c(R_1(i)) + c(R_2(i)) \right). \quad (6.22)
\]

This cost function 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

\[
c_{\text{vert}}^{(2)} := \min_{i_1 < i < i_2} \max(c(R_1(i)), c(R_2(i))). \quad (6.23)
\]

Usage of this cost function induces more splitting to take place, as it is always less than or equal to the cost function in (6.22). Cost function (6.23) has the disadvantage that the two finer grids that will be created can differ greatly in size: the minimum of \( c(R_1(i)) \) and \( c(R_2(i)) \) can be much smaller than the maximum of the two. To remedy this unbalance, we propose the cost function

\[
c_{\text{vert}}^{(3)} := \min_{i_1 < i < i_2} \left( 2 \max(c(R_1(i)), c(R_2(i))) - \min(c(R_1(i)), c(R_2(i))) \right). \quad (6.24)
\]

In our numerical examples, we will use cost function (6.24). 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. The overlap added here is at the top and bottom of the two smaller subgrids though.

From the three options

- use the smallest enclosing rectangle \( R \);
- split \( R \) vertically along the index which gives the lowest cost and use the smallest enclosing rectangles in \( R_1 \) and \( R_2 \);
- split \( R \) horizontally along the index which gives the lowest cost and use the smallest enclosing rectangles in \( R_1 \) and \( R_2 \),

we choose the one with the lowest cost. This choice is overridden if the number of nodes necessary to refine \( R \) entirely is less than twice the total number of nodes of the coarser
grid G that we are applying the domain decomposition algorithm to. If this holds, we always refine R entirely.

If we decide to split R either vertically or horizontally, we repeat the domain decomposition algorithm recursively for the smallest enclosing rectangles in R1 and R2. These recursive calls make yet another subdivision of the grids possible. Hence, the original rectangle R can be split in many smaller rectangles. Due to this recursion, one call to the domain decomposition algorithm causes a branching each time the algorithm decides to split a rectangle. Every branch ends when the option of using the smallest enclosing rectangle is chosen. There is no pre-set maximum recursion depth. Note that there is always an upper bound for the recursion depth in practice: if R is split in such a way that every flagged box lies in a separate rectangle, no further splitting will occur.

The amount of overlap that is being added to the parts when splitting a rectangle is registered. The reason for registering the overlap added to a subgrid is that the overlap zone should not be refined further. If we would not take this precaution, larger and larger parts of the grids could be refined on finer levels, which is both unnecessary and undesirable.

In the LDC algorithm, we have already seen that one grid point may lie in more than one grid. Thus far, a grid point could only lie in grids with different grid sizes and it was apparent that the approximation on the finest grid should be considered most accurate. Due to the overlap used in the domain decomposition algorithm, there may be two or more grids at one level that contain the same grid point. To decide which approximation should be considered most accurate, we keep track of a calculation history. If a grid point lies in more than one grid at the finest level with grids that contain the grid point, then we use the approximation calculated most recently. In our calculations the solutions on the different grids are stored in separate files. Furthermore an entry is added to a list each time a calculation takes place on a (sub)grid. The solution files together with the list provide the composite grid approximation.

When there is more than one grid at a certain level, it may be necessary to solve the discrete problem more than once on each subgrid. In order to let overlapping subgrids communicate, we calculate approximations on them in turns. For two overlapping grids, we use the most recently calculated approximations in one grid as a boundary condition for that part of the boundary of the other grid that lies in the first grid due to overlap. Calculations on grids at the same level that are not connected, are completely independent and may therefore be performed in parallel.

The composite grids constructed in the multi-level LDC algorithm extended with domain decomposition may conveniently be represented by a tree, see Figure 6.4. The algorithm itself, for the composite grid of Figure 6.4 is shown in Figure 6.5. The algorithm can be formulated as follows.

**Algorithm 6.3**

*Multi-level LDC algorithm with domain decomposition*

*Initialization*
6.3 Domain decomposition

Figure 6.4: 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; these three grids are labeled ‘base.1’, ‘base.2’, and ‘base.3’. There are four grids at Level 2.

- Solve the basic coarse grid problem (3.11) (the Level 0 grid problem).
- For $l = 0, 1, \ldots, l_{\text{max}} - 1$, do
  - For each grid $G$ at Level $l$ do
    * Determine the location of the high activity area in grid $G$ and choose finer Level $l + 1$ grids in this area.
    * For each subgrid of $G$, interpolate the current composite grid solution to the subgrid and solve the problem on the subgrid at Level $l + 1$.
  - Perform domain decomposition iterations (i.e. solve again on all subgrids) at Level $l + 1$ if there are overlapping grids.

**Iteration, $i = 1, 2, \ldots$**

- For $l = l_{\text{max}} - 1, l_{\text{max}} - 2, \ldots, 0$, do
  - For each grid $G$ at Level $l$ do a coarse grid correction solve on $G$.
  - Perform domain decomposition iterations at Level $l$ if there are overlapping grids.
- For $l = 0, 1, \ldots, l_{\text{max}} - 1$, do
  - For each grid $G$ at Level $l + 1$, solve on $G$ with new boundary conditions; reuse defects calculated during the previous correction solve if this is not the finest level.
  - Perform domain decomposition iterations (i.e. solve again on all subgrids) at Level $l + 1$ if there are overlapping grids.

Berger and Colella [Berger and Colella, 1989] consider a local adaptive grid refinement method for hyperbolic conservation laws. In their approach, called the adaptive mesh refinement (AMR) algorithm, the refined region consists of a number of rectangular grid
patches too. 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 make use of a buffer zone around high activity boxes. Their method to create the rectangular fine grids is heuristic, like ours. Berger and Colella form the smallest rectangle enclosing all flagged boxes and measure its efficiency by the ratio of all flagged cells to the total number of cells. If the efficiency is less than a given minimum efficiency, 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 [Berger and Oliger, 1984] present a similar adaptive grid refinement algorithm, but their method allows for the usage of rectangles of arbitrary orientation. By allowing the arbitrary orientation, it is possible to have a coordinate system which is locally approximately aligned with some feature in the solution. The AMR algorithm is generalized to three space dimensions in [Bell et al., 1994].

6.4 An example problem

To illustrate the working of the LDC algorithm combined with domain decomposition, we consider the following model problem which is taken from [Bennett and Smooke, 1999, Valdati, 1997]:

\[
\begin{aligned}
& -\Delta u + \frac{\partial u}{\partial x} + \frac{\partial u}{\partial y} = f, \\
& \quad \text{in } \Omega = (0,1) \times (0,4), \\
& u = g, \\
& \quad \text{on } \partial \Omega.
\end{aligned}
\]

In (6.25), \( f \) and \( g \) have been chosen such that

\[
u(x,y) = \tanh[\alpha(-4x - 2y + 3)] + 1, \quad \alpha = 5.
\]
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. The analytical solution is shown in Figure 6.6.

For the discretization of (6.25), we choose a uniform coarse grid \( \Omega^H \) in \( \Omega \) with grid sizes \( \Delta x = \Delta y = H = 1/5 \). Both the diffusion and the convection term in (6.25) are discretized using standard central differences on all grids. We carry out five different experiments; the difference between these runs is the maximum level of refinement in the LDC algorithm. In all experiments, the threshold value \( \epsilon \) for the weight function is set to 2 at all levels and the layering parameter is chosen according to (6.14), (6.15) with \( N_{\text{layer}(0)} \) equal to 3. The refinement factor \( \sigma \) equals 2. The overlap parameter \( N_{\text{dd}} \) for the domain decomposition algorithm is set to 1. At levels with more than one grid, the problems on the subgrids are solved one after another. Two solves take place on each subgrid before proceeding to the next level.

Figure 6.7 shows the composite grid and the error of the LDC solution for the experiment with \( l_{\text{max}} = 3 \); the tree representation of the grid hierarchy corresponding to this experiment is shown in Figure 6.8. Figure 6.9 shows the subgrids at the finest level and the error of the solution on these subgrids for the experiment with \( l_{\text{max}} = 4 \).

For comparison reasons, approximations have been calculated on uniform grids with grid sizes equal to the grid sizes of the finest grids of the LDC composite grids. All numerical results are listed in Table 6.2. As before, the errors listed are maximum errors and the number of LDC iterations is set to 1. We find that the error on the LDC composite grids is practically equal to the error on the equivalent uniform grid for all experiments. Only in the last experiment when \( l_{\text{max}} = 5 \), we see that the error of the LDC solution does not decrease. This can be explained as follows: the steep gradient of the solution can be well resolved on the LDC composite grid with \( l_{\text{max}} = 4 \). The maxi-
Figure 6.7: The composite grid and the error of the LDC solution for the experiment with $l_{\text{max}} = 3$. The tree representation of this composite grid is shown in Figure 6.8.

The composite grid is not located within the Level 4 grid. It is located at a grid point at Level 1 outside the area that would be refined by introducing more levels of refinement. For this reason, adding the additional Level 5 grid does not decrease the (maximum) error on the composite grid.

The entries in the fourth columns of Tables 6.2(a) and 6.2(b) are the execution times of the calculations in seconds. Again, we listed the average of the solution times of ten identical runs for each experiment. The execution times of Table 6.2(b) may seem disappointing at first sight. It should be noted however that we have not used any parallelization to produce these execution times. Rather than solving the problems on nonoverlapping subgrids at a level in parallel, we have solved them sequentially. On the Level 4 grid, e.g., the problems on the 8 subgrids have been solved one after another.
6.4 An example problem

Figure 6.8: Tree representation of the grid hierarchy for the experiment with $l_{\text{max}} = 3$. The composite grid itself is shown in Figure 6.7.

Table 6.2: Maximum norms of the error and execution times of the numerical experiments for problem (6.25).

(a) Uniform grids. (b) LDC composite grids.

<table>
<thead>
<tr>
<th>$H$</th>
<th>Points</th>
<th>Error</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/5</td>
<td>126</td>
<td>$2.45 \cdot 10^{-3}$</td>
<td>0.07</td>
</tr>
<tr>
<td>1/10</td>
<td>451</td>
<td>$3.43 \cdot 10^{-3}$</td>
<td>0.12</td>
</tr>
<tr>
<td>1/20</td>
<td>1,701</td>
<td>$9.43 \cdot 10^{-2}$</td>
<td>0.36</td>
</tr>
<tr>
<td>1/40</td>
<td>6,001</td>
<td>$1.94 \cdot 10^{-2}$</td>
<td>1.76</td>
</tr>
<tr>
<td>1/80</td>
<td>26,001</td>
<td>$4.66 \cdot 10^{-3}$</td>
<td>10.63</td>
</tr>
<tr>
<td>1/160</td>
<td>103,201</td>
<td>$3.00 \cdot 10^{-3}$</td>
<td>59.53</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$l_{\text{max}}$</th>
<th>Grid points</th>
<th>Error</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>126</td>
<td>$2.45 \cdot 10^{-4}$</td>
<td>0.08</td>
</tr>
<tr>
<td>1</td>
<td>+231 = 357</td>
<td>$3.43 \cdot 10^{-4}$</td>
<td>0.69</td>
</tr>
<tr>
<td>2</td>
<td>+821 = 1,181</td>
<td>$9.42 \cdot 10^{-4}$</td>
<td>2.65</td>
</tr>
<tr>
<td>3</td>
<td>+2,480 = 3,661</td>
<td>$1.92 \cdot 10^{-2}$</td>
<td>7.13</td>
</tr>
<tr>
<td>4</td>
<td>+7,328 = 10,989</td>
<td>$5.02 \cdot 10^{-3}$</td>
<td>20.14</td>
</tr>
<tr>
<td>5</td>
<td>+22,778 = 33,767</td>
<td>$5.02 \cdot 10^{-3}$</td>
<td>54.70</td>
</tr>
</tbody>
</table>

The LDC algorithm with domain decomposition outperforms the uniform grid solver by far with respect to memory usage. In the case where the uniform grid solver solves the discretized boundary value problem on a grid with 103,201 grid points, the LDC composite grid has a total number of nodes of 33,767 that is composed of 32 subgrids. The largest constituent subgrid has only 1,763 nodes and is located at Level 5; see Table 6.3 which gives detailed information on the LDC composite grids for this example problem. This implies that calculations take place on grids with 1,763 grid points or less only, which is a huge saving over the single uniform grid that has 103,201 points. Apart from low memory usage, a reduction of the number of grid points may also mean faster convergence of the solver used on the individual tensor-product grids.
Adaptive multi-level LDC in combination with domain decomposition

Figure 6.9: The subgrids at the finest level and the error of the LDC solution for the experiment with $l_{\text{max}} = 4$.

6.5 Regridding

Refraining a grid and solving the boundary value problem under consideration on the new composite grid may cause the area of high activity of the solution to move. This phenomenon was for instance observed for the Bunsen flame problem in [Bennett and Smooke, 1998] by Bennett and Smooke. When the area of high activity moves, areas of the grid may be refined unnecessarily or areas may not be refined whereas they do require refinement. Therefore, regridding is used before proceeding to a finer level.

We follow the regridding procedure from [Bennett and Smooke, 1998]. Rather than proceeding directly from Level 1 to Level 1 + 1, we use the following strategy. First, we store the current composite grid solution. This solution is projected onto the Level 0 grid. Based on this projected solution, we choose new Level 1 grids. The locations of the subgrids are found by calculating the weight function and using the domain decomposition algorithm. Next, the stored composite grid solution is projected on the new Level 1 grids and based on these approximations, Level 2 grids are chosen. This procedure continues until we have chosen the Level 1 + 1 grids.
6.5 Regridding

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

Table 6.3: Detailed information on the LDC composite grids for problem (6.25).

The complete LDC algorithm featuring both domain decomposition and regridding is illustrated in Figure 6.10. We do not present an example of the regridding procedure in this chapter; the full algorithm will however be used in the next chapter in our numerical simulations of the Bunsen flame problem. The multi-level LDC algorithm with domain decomposition and regridding can be formulated as follows.

Algorithm 6.4
Multi-level LDC algorithm with domain decomposition and regridding

Initialization

- Solve the basic coarse grid problem (3.11) (the Level 0 grid problem).
- For \( l = 0, 1, \ldots, l_{\text{max}} - 1 \), do
  - For each grid \( G \) at Level \( l \) do
    * Determine the location of the high activity area in grid \( G \) and choose finer Level \( l + 1 \) grids in this area.
For each subgrid of \( G \), interpolate the current composite grid solution to the subgrid and solve the problem on the subgrid at Level \( l + 1 \).

- Perform domain decomposition iterations at Level \( l + 1 \) if there are overlapping grids.

- If the current Level \( l \) is less than \( l_{\text{max}} - 1 \), perform regridding:
  * Store the current composite grid solution.
  * Project the composite grid solution on the basic coarse grid.
  * For \( l' = 0, 1, \ldots, l \), do
    - For each grid \( G' \) at Level \( l' \) do
      - Determine the location of the high activity area in \( G' \) and choose finer Level \( l' + 1 \) grids in this area.
      - For each subgrid of \( G' \), interpolate the stored composite grid solution to the subgrid at Level \( l' + 1 \) (do not solve).

* **Iteration, \( i = 1, 2, \ldots \)**

  - For \( l = l_{\text{max}} - 1, l_{\text{max}} - 2, \ldots, 0 \), do
    - For each grid \( G \) at Level \( l \), do a coarse grid correction solve on \( G \).
    - Perform domain decomposition iterations at Level \( l \) if there are overlapping grids.

  - For \( l = 0, 1, \ldots, l_{\text{max}} - 1 \), do
    - For each grid \( G \) at Level \( l + 1 \), solve on \( G \) with new boundary conditions; reuse defects calculated during the previous correction solve if this is not the finest level.
    - Perform domain decomposition iterations at Level \( l + 1 \) if there are overlapping grids.
Chapter 7

Numerical simulation of the Bunsen flame

7.1 Discretization of the partial differential equations

We recall the system of partial differential equations that forms the mathematical model for the Bunsen flame problem. The equations were presented in Chapter 2, cf. (2.35), (2.38), (2.39), (2.40), and (2.42). The number of dependent variables in the Bunsen flame problem is 9, namely 5 mass fractions, \(Y_{CH_4}, Y_{O_2}, Y_{H_2O}, Y_{CO_2},\) and \(Y_{N_2}\), the radial velocity \(v_r\), the axial velocity \(v_z\), the vorticity \(\omega\), and the temperature \(T\). These variables satisfy the following conservation laws:

\[
\rho v_r \frac{\partial Y_i}{\partial r} + \rho v_z \frac{\partial Y_i}{\partial z} = \frac{1}{Le_i} \left\{ \frac{1}{r} \frac{\partial}{\partial r} \left( r \lambda \frac{\partial Y_i}{\partial r} \right) + \frac{\partial}{\partial z} \left( \frac{\lambda}{c_p} \frac{\partial Y_i}{\partial z} \right) \right\} + w_i, \tag{7.1}
\]

\[
\frac{\partial^2 v_r}{\partial r^2} + \frac{\partial^2 v_r}{\partial z^2} = \frac{\partial \omega}{\partial z} - \frac{1}{r} \frac{\partial v_r}{\partial r} + \frac{v_r}{r^2} - \frac{\partial}{\partial r} \left( \nu \cdot \nabla \rho \right), \tag{7.2}
\]

\[
\frac{\partial^2 v_z}{\partial r^2} + \frac{\partial^2 v_z}{\partial z^2} = - \frac{\partial \omega}{\partial r} - \frac{1}{r} \frac{\partial v_r}{\partial z} - \frac{\partial}{\partial z} \left( \nu \cdot \nabla \rho \right), \tag{7.3}
\]

\[
\frac{\partial^2}{\partial r^2} \left( \mu \omega \right) + \frac{\partial^2}{\partial z^2} \left( \mu \omega \right) = - \frac{\partial}{\partial r} \left( \frac{\mu \omega}{r} \right) + \rho v_r \frac{\partial \omega}{\partial r} + \rho v_z \frac{\partial \omega}{\partial z} - \frac{1}{r} \rho v_r \omega + \nabla \rho \cdot \nabla \left( \frac{v_r v_z}{2} \right) - \nabla \rho \cdot g, \tag{7.4}
\]

\[
c_p \left( \rho v_r \frac{\partial T}{\partial r} + \rho v_z \frac{\partial T}{\partial z} \right) = \frac{1}{r} \frac{\partial}{\partial r} \left( r \lambda \frac{\partial T}{\partial r} \right) + \frac{\partial}{\partial z} \left( \frac{\lambda}{c_p} \frac{\partial T}{\partial z} \right) - \sum_{i=1}^{N-1} h_i^* w_i. \tag{7.5}
\]
Note that (7.1) is only solved for four of the five mass fractions; the mass fraction of nitrogen, the inert species, follows from condition (2.3), which states that the five mass fractions sum up to unity.

Due to the nature of the LDC method, the partial differential equations (7.1)–(7.5) need to be discretized on tensor-product grids only. The initial coarse grid is nonuniform; it is chosen to be more finely spaced in the region above the inner jet, because it is known that the flame will be formed in that area. The exact \( r \) and \( z \)-coordinates of the coarse initial tensor-product grid are listed in Table 7.1. The lower part of the grid is shown in Figure 7.1.

To discretize the system of differential equations on the initial coarse grid and on local finer grids, we use finite difference discretizations. Bennett and Smooke [Bennett and Smooke, 1998] list some of the discretizations that are applied; a complete overview of the finite difference formulas is listed in [Valdati, 1997, App. III]. Here, we will give the formulas for interior points and for derivatives with respect to the radial coordinate as well as mixed derivatives. Axial derivatives are treated similarly. We apply the following standard stencils (\( \varphi_{ij} \) denotes the approximation of the value of \( \varphi \) in grid point \( (r_i, z_j) \)):

\[
\begin{align*}
\frac{\partial \varphi}{\partial r} & \bigg|_{r_i, z_j} = \frac{\varphi_{i+1,j} - \varphi_{i-1,j}}{r_{i+1} - r_{i-1}}, \\
\frac{\partial^2 \varphi}{\partial r^2} & \bigg|_{r_i, z_j} = \frac{2\varphi_{i+1,j} - \varphi_{i-1,j}}{(r_{i+1} - r_i)(r_{i+1} - r_{i-1})} - \frac{2\varphi_{ij}}{(r_{i+1} - r_i)(r_i - r_{i-1})} \\
& \quad + \frac{2\varphi_{i-1,j}}{(r_i - r_{i-1})(r_{i+1} - r_{i-1})}, \\
\frac{\partial^2 \varphi}{\partial r \partial z} & \bigg|_{r_i, z_j} = \frac{\varphi_{i+1,j+1} - \varphi_{i+1,j-1} - \varphi_{i-1,j+1} + \varphi_{i-1,j-1}}{(r_{i+1} - r_i)(z_{j+1} - z_{j-1})}, \\
\frac{\partial}{\partial r} \left( a \frac{\partial \varphi}{\partial r} \right) & \bigg|_{r_i, z_j} = \frac{2a_{i+1/2,j}(\varphi_{i+1,j} - \varphi_{ij})}{(r_{i+1} - r_i)(r_{i+1} - r_{i-1})} - \frac{2a_{i-1/2,j}(\varphi_{ij} - \varphi_{i-1,j})}{(r_i - r_{i-1})(r_{i+1} - r_{i-1})}, \\
\frac{\partial}{\partial z} \left( a \frac{\partial \varphi}{\partial z} \right) & \bigg|_{r_i, z_j} = \frac{a_{i+1/2,j}(\varphi_{i+1,j+1} + \varphi_{i,j+1} - \varphi_{i+1,j-1} - \varphi_{i,j-1})}{(r_{i+1} - r_i)(z_{j+1} - z_{j-1})} \\
& \quad - \frac{a_{i-1/2,j}(\varphi_{ij+1} + \varphi_{i,j+1} - \varphi_{ij-1} - \varphi_{ij-1})}{(r_{i+1} - r_i)(z_{j+1} - z_{j-1})}.
\end{align*}
\]

Special care is taken of the convective terms in the species continuity equation (7.1), the vorticity equation (7.4), and the energy equation (7.5). Convective terms are those terms of the form (\( \varphi = Y_i, \omega, T \))

\[
\rho v_r \frac{\partial \varphi}{\partial r} + \rho v_z \frac{\partial \varphi}{\partial z}.
\]

These terms are discretized using an upwind discretization, viz.

\[
\left. v_r \frac{\partial \varphi}{\partial r} \right|_{r_i, z_j} = \alpha_{ij} \frac{\varphi_{i+1,j} - \varphi_{ij}}{r_{i+1} - r_i} + \beta_{ij} \frac{\varphi_{ij} - \varphi_{i-1,j}}{r_i - r_{i-1}},
\]

where \( \alpha_{ij} \) and \( \beta_{ij} \) are constants chosen to satisfy the upwind condition.
in which the coefficients $\alpha_{ij}$ and $\beta_{ij}$ are defined as

$$\alpha_{ij} := \min \left\{ \frac{(v_r)_{i+1,j} + (v_r)_{ij}}{2}, 0 \right\}, \quad \beta_{ij} := \max \left\{ \frac{(v_r)_{ij} + (v_r)_{i-1,j}}{2}, 0 \right\}. \quad (7.13)$$

The dependent variables approach constant values and are therefore slowly varying functions at the outer radial and at the outflow boundaries. For this reason, 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 [Bennett and Smooke, 1998, Sec. 4.2]. Especially the boundary condition for the axial velocity at the symmetry axis, viz,

$$\frac{\partial v_z}{\partial r} = 0, \quad (7.14)$$

needs to be treated in a special way. We use the Taylor series expansion

$$v_z(r + \Delta r, z) = v_z(r, z) + \Delta r \frac{\partial v_z}{\partial r}(r, z) + \frac{1}{2} (\Delta r)^2 \frac{\partial^2 v_z}{\partial r^2}(r, z) + O((\Delta r)^3). \quad (7.15)$$

The second term at the right hand side vanishes due to boundary condition (7.14); for the third term we use the conservation law (7.3) for $v_z$. This leads to [Bennett and Smooke, 1998, form. (18)]:

$$v_z|_{r_{i+1}, z_j} = v_z|_{r_i, z_j} - \frac{(r_{i+1} - r_i)^2}{2} \left( \frac{\partial^2 v_z}{\partial z^2} + \frac{\partial \omega}{\partial r} + \frac{\partial}{\partial z} \left( \frac{v_z}{\rho} \frac{\partial \rho}{\partial z} \right) + \frac{\partial^2 v_r}{\partial r \partial z} \right). \quad (7.16)$$

This equation can be discretized to higher order than (7.14).

Discretization of the partial differential equations yields a large system of coupled nonlinear equations. The number of equations in the discrete system equals the number of dependent variables (which is 9 for the Bunsen flame problem) multiplied by the number of grid points. In the next section we will detail the solution process used to solve the system.

### 7.2 Solving the discrete system

To linearize the discrete system of equations obtained by discretization of the partial differential equations, we use (a damped form of) Newton’s method. We will denote the discrete system of nonlinear equations by

$$F(U) = 0, \quad (7.17)$$

in which $U$ is the vector with all unknowns. Newton’s method with damping can be formulated as the iteration ($n = 0, 1, 2, \ldots$)

$$J(U_n) (U_{n+1} - U_n) = -\lambda_n F(U_n). \quad (7.18)$$
Numerical simulation of the Bunsen flame

Table 7.1: The radial ($r$) and axial ($z$) coordinates (in centimeters) of the initial coarse tensor-product grid for the Bunsen flame problem.

<table>
<thead>
<tr>
<th>Radial coordinate</th>
<th>Axial coordinate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Range</td>
<td>Grid size</td>
</tr>
<tr>
<td>0–0.76</td>
<td>0.01</td>
</tr>
<tr>
<td>-0.80</td>
<td>0.02</td>
</tr>
<tr>
<td>-0.90</td>
<td>0.025</td>
</tr>
<tr>
<td>-1.00</td>
<td>0.05</td>
</tr>
<tr>
<td>-1.40</td>
<td>0.1</td>
</tr>
<tr>
<td>-2.00</td>
<td>0.2</td>
</tr>
<tr>
<td>-3.00</td>
<td>0.25</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In (7.18), $J$ denotes the Jacobi matrix of the system, defined by

$$J(U) := \frac{\partial F(U)}{\partial U},$$

(7.19)

and $\lambda_n$ is a damping factor satisfying $\lambda_n \in (0, 1]$. The Jacobi matrix is calculated numerically. Given an initial guess $U_0$, (7.18) produces a new approximation $U_1$ for the solution $U$ of system (7.17). The approximation $U_1$ is used to find $U_2$, and so on. A well-known property of Newton’s method is that it converges quadratically for an initial guess $U_0$ that is close to the actual solution $U$. This convergence is only local though; if the initial guess $U_0$ is outside the convergence domain of the solution $U$, Newton’s method may either converge to a wrong solution or it may not converge at all. In our simulations, the Newton iteration is said to have converged if the 2-norm of the scaled correction vector $U_{n+1} - U_n$ is smaller than a given tolerance.

To bring an initial guess within the convergence domain, a pseudo time-stepping technique is applied. A number (typically 50) of adaptively chosen time steps is taken to find an initial guess for the Newton iteration to solve the steady state equations. This time-stepping procedure is considered in detail in [Bennett and Smooke, 1998, Smooke et al., 1989, Valdati, 1997]. After the time-stepping process, the steady state equations are solved with increasing accuracy. Four successive solves take place. The tolerance of these Newton solves are $8 \cdot 10^{-4}$, $4 \cdot 10^{-4}$, $2 \cdot 10^{-4}$, and $1 \cdot 10^{-4}$, respectively. This implies that the final solution has an accuracy of $1 \cdot 10^{-4}$.

In each Newton step, the update $U_{n+1} - U_n$ to the current approximation $U_n$ has to be found by solving a large linear system, see (7.18). This system is solved using the biconjugate gradient stabilized method (Bi-CGSTAB) due to Van der Vorst [van der Vorst, 1992]. The Bi-CGSTAB method is also treated in e.g. [Saad, 1996, Sec. 7.4.2]. The method is said to have converged if the 2-norm of the scaled residual vector is smaller than one tenth of the tolerance of the Newton iteration. A Gauss-Seidel preconditioner is used to speed up convergence. To decrease the computational costs of computing Jacobi matrices, the Jacobi matrix is kept constant during (at most) five steps of the time-stepping process. The Jacobi matrix may have to be evaluated more frequently depending on the convergence behavior of the iteration. For details we refer to [Bennett and Smooke, 1998].
7.3 Numerical results

Figure 7.1: Lower part of the initial tensor-product grid for the Bunsen flame problem. The actual grid extends from $z = 0$ cm till $z = 25$ cm.

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 an already converged solution. This starting solution was calculated by Bennett and Smooke [Bennett and Smooke, 1998]. They construct a crude initial guess by using the prescribed boundary conditions at the inflow boundary and by duplicating these at horizontal levels. Via time-stepping, this initial guess is brought in the convergence domain of Newton’s method. For the local fine grids used in the LDC method, an initial guess is readily found using interpolation of approximations that have been found on coarser grids.

7.3 Numerical results

Four different LDC simulations have been carried out. Each simulation is started with an already converged solution on the base tensor-product grid as described in Section 7.1. We use the multi-level LDC algorithm with domain decomposition and regridding as formulated in the previous sections. 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. During the first solve on a grid, time-stepping is used to bring the initial guess within the Newton convergence domain. We use 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 do-
main decomposition iterations are carried out to improve the boundary conditions at the internal interfaces. During the domain decomposition iteration, no time-stepping is employed.

For the domain decomposition part of our LDC simulations we have used the following parameter settings. The flagging of high activity boxes as discussed in Section 6.1 is based on the smoothness of the methane mass fraction $Y_{CH_4}$. The number of smoothing passes for the weight function is set to 10. The parameter $\alpha$ in the definition of the weight function, cf. (6.7), is chosen as $1 \cdot 10^6$. The threshold value $\epsilon$ for flagging boxes equals 2 at all levels. The number of neighboring boxes that are flagged, $N_{layer}$, is 10 for the initial tensor-product grid. The refinement factor $\sigma$ is 2. On finer grids, the layering parameter $N_{layer}$ is chosen according to (6.14), (6.15). The amount of overlap in the domain decomposition algorithm is the same on all levels; it is set to 5.

Plots of all dependent variables that have been calculated are shown in Figure 7.3. These results have been obtained on a composite grid based on the tensor-product grid from Table 7.1 with two additional levels of refinement. The first level consists of two subgrids; the second level has four subgrids. It is clear that all dependent variables except for the nitrogen mass fraction have large gradients in the reaction zone. A remarkable characteristic of the Bunsen flame problem is that the structure of the flame is similar on the grids with different resolutions, but that its size increases on the finer grids. The increase is strongest when the first level of refinement is added. The flame length still increases, but less rapidly, with additional refinement. This is illustrated in Figure 7.4, which shows the projection of $Y_{CH_4}$ on the LDC composite grids. We will analyze this phenomenon at the end of this section.

The Bunsen flame problem was previously treated by Bennett [Valdati, 1997] and Bennett and Smooke [Bennett and Smooke, 1998]. In both references, simulations have been performed with the local rectangular refinement (LRR) method. In the LRR method, an unstructured grid is constructed from an initial tensor-product grid by flagging and refining high activity boxes. The restriction that all grid lines should extend from one domain boundary to the other, as is the case for tensor-product grids, is lifted. At interior boundary points, i.e., points at interfaces between coarse and fine grids, special discretization stencils are applied. Next the problem is solved on the complete unstructured grid.

An important difference between the LRR algorithm and the LDC algorithm is that all equations at all grid points are solved simultaneously in the LRR approach. In the LDC algorithm with domain decomposition, the equations 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. Secondly overlapping subgrids can be used at one level of refinement. The approximations on the subgrids at one level are coupled again via the domain decomposition iteration.

Because the composite grid in the LRR algorithm is unstructured, only high activity boxes and their neighboring boxes are refined. In the LDC approach, we have chosen to use rectangular grids only. Although this choice may cause the refinement of un-
Table 7.2: 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.

<table>
<thead>
<tr>
<th>Level</th>
<th>Grid</th>
<th>Grid points</th>
<th>Sum</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>bunsenfine</td>
<td>9,408</td>
<td>9,408</td>
<td>9,408</td>
</tr>
<tr>
<td>1</td>
<td>bunsenfine.1</td>
<td>7,085</td>
<td>7,161</td>
<td>14,246</td>
</tr>
<tr>
<td></td>
<td>bunsenfine.2</td>
<td>7,161</td>
<td>7,085</td>
<td>14,246</td>
</tr>
<tr>
<td>2</td>
<td>bunsenfine.1.1</td>
<td>10,075</td>
<td>10,125</td>
<td>20,200</td>
</tr>
<tr>
<td></td>
<td>bunsenfine.1.2</td>
<td>10,125</td>
<td>10,075</td>
<td>20,200</td>
</tr>
<tr>
<td></td>
<td>bunsenfine.2.1</td>
<td>10,375</td>
<td>10,375</td>
<td>20,750</td>
</tr>
<tr>
<td></td>
<td>bunsenfine.2.2</td>
<td>10,201</td>
<td>10,375</td>
<td>20,576</td>
</tr>
<tr>
<td>3</td>
<td>bunsenfine.1.1.1</td>
<td>16,303</td>
<td>16,303</td>
<td>32,606</td>
</tr>
<tr>
<td></td>
<td>bunsenfine.1.1.2</td>
<td>16,065</td>
<td>16,065</td>
<td>32,130</td>
</tr>
<tr>
<td></td>
<td>bunsenfine.1.2.1</td>
<td>14,673</td>
<td>14,673</td>
<td>29,346</td>
</tr>
<tr>
<td></td>
<td>bunsenfine.1.2.2</td>
<td>14,553</td>
<td>14,553</td>
<td>29,106</td>
</tr>
<tr>
<td></td>
<td>bunsenfine.2.1.1</td>
<td>15,355</td>
<td>15,355</td>
<td>30,710</td>
</tr>
<tr>
<td></td>
<td>bunsenfine.2.1.2</td>
<td>15,189</td>
<td>15,189</td>
<td>30,378</td>
</tr>
<tr>
<td></td>
<td>bunsenfine.2.2.1</td>
<td>16,653</td>
<td>16,653</td>
<td>33,306</td>
</tr>
<tr>
<td></td>
<td>bunsenfine.2.2.2</td>
<td>16,539</td>
<td>16,539</td>
<td>33,078</td>
</tr>
<tr>
<td></td>
<td>bunsenfine.2.2.3</td>
<td>16,539</td>
<td>16,539</td>
<td>33,078</td>
</tr>
</tbody>
</table>

Table 7.3: Detailed information on the LDC composite grid for the Bunsen flame problem with maximum level of refinement 3.

flagged boxes, i.e., of boxes that do not actually need refinement, it has the advantage that we can use simple data structures as well as standard discretization stencils at all grid points. The usage of rectangular grids in the LDC algorithm explains the difference in the total number of grid points used in the LDC simulations and in the LRR simulations, cf. Table 7.2. This number is larger for the LDC simulations. In the LDC simulations, many small grids are used rather than one large grid however. As shown in Table 7.3 the largest tensor-product grid in the LDC simulation with three additional levels of refinement is located at the maximum level and has only 16,653 grid points, which is a large saving over the single tensor-product grid of 312,872 grid points as well as over the LRR grid of 126,393 grid points. Although the LDC method saves memory, it proves more expensive with respect to CPU time than the LRR algorithm. This can be attributed to the strong coupling of the equations governing the Bunsen flame.

To verify the LRR results, Bennett and Smooke have created equivalent tensor-product grids for each of the unstructured LRR grids by extending all grid lines to the domain boundaries. The Bunsen flame problem has been solved on these tensor-product grids too. Both the LRR results and the equivalent tensor-product results are similar to our LDC results. In particular, the increase of the size of the Bunsen flame was also observed in Bennett and Smooke [1998].
The size of a flame can be expressed via the flame length \( L_f \), see Figure 2.1. There are various definitions of the flame length. We follow Bennett and Smooke in order to compare our results to theirs, and define \( L_f \) as the minimum value of the axial coordinate on the \( z \)-axis for which the methane mass fraction \( Y_{CH_4} \) is less than \( 1 \cdot 10^{-4} \). In Table 7.2, we list the values of the flame length found in our LDC calculations (first column) and we quote the results for the LRR simulations (second column) as well as those for the equivalent tensor-product grids (third column). The numbers for the second and third column have been taken from [Bennett and Smooke, 1998, Table 2]. Bennett and Smooke have also performed LRR simulations in which the adaptation of the grid is not only based on the methane mass fraction, but on the temperature or on all other dependent variables too, see [Bennett and Smooke, 1998, Tables 3 and 4]. A similar dependence on the size of the flame on the grid resolution was found in these experiments.

Note that the first entry in the LDC column in Table 7.2 is blank. This is because the LDC simulations have been started with a finer initial tensor-product grid than the LRR simulations. The initial LRR tensor-product grid proved to be too coarse to adequately represent the coarse grid solution in the LDC algorithm after it had been updated by an LDC correction solve. The initial tensor-product grid used in the LDC simulations has a resolution that corresponds with the LRR grid that has one level of refinement. The last entry in the third column is also missing. The reason is that the tensor-product grid corresponding to this entry has too many grid points, so that the problem exceeds the available computer memory, see [Bennett and Smooke, 1998].

The strong dependence of the flame length on the size of the grid is explained in [Bennett and Smooke, 1998]. The authors show, that the laminar flame speed \( v_{lam} \) is proportional to the square root of the mixture averaged diffusion coefficient \( D \) [Bennett and Smooke, 1998, form. (20)]. The first order upwind discretization of the convective terms in the conservation laws, cf. (7.11), (7.12), adds artificial diffusion. Due to the artificial diffusion, \( D \) is overestimated. Refinement of the grid causes less artificial diffusion, and hence a lower diffusion coefficient \( D \) and a lower flame speed. Due to the balance of velocities in a laminar flame, a lower flame speed causes the flame to rise. This can be understood as follows. In Figure 7.2(a) (this figure corresponds with [Bennett and Smooke, 1998, Figure 10]), we see that a lower \( v_{lam} \) causes the angle \( \gamma \) between the \( z \)-axis and the flame front to shrink. As the radius \( r \) of the base of the flame is fixed, a reduction of \( \gamma \) causes an increase of the flame length \( L_f \), see Figure 7.2(b).

In addition to this qualitative explanation by Bennett and Smooke for the grid dependency of the flame length, we will carry out a small numerical experiment supporting their analysis. To this end, we set

\[
D(H) = D + D_{art}(H),
\]

with \( D(H) \) the diffusion coefficient on a grid with typical grid size \( H \), \( D \) the real diffusion coefficient, and \( D_{art}(H) \) the artificial diffusion introduced by the upwind scheme. The latter diffusion coefficient is proportional to the grid size \( H \), viz.

\[
D_{art}(H) \doteq CH,
\]

with \( C \) constant. Expression (7.21) shows that the artificial diffusion tends to zero if the
7.3 Numerical results

grid size $H$ tends to zero. Hence, the artificial diffusion is smaller on finer grids, causing the diffusion coefficient $D(H)$ to tend to $D$, cf. (7.20). Note that usage of a different discretization scheme than the standard upwind scheme applied here may also reduce the amount of artificial diffusion. We expect the flame lengths to be more accurate on the coarser grids if we would use e.g. the modified Thiart scheme by Van ’t Hof et al. [van ’t Hof et al., 1998]. As stated, the laminar flame speed, which we denote by $v_{\text{lam}}$ is proportional to $\sqrt{D(H)}$, so that (cf. [Bennett and Smooke, 1998, form. (20)])

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

(7.22)

with $\alpha$ constant. For the angle $\gamma$ between the z-axis and the flame front, we have

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

(7.23)

In (7.23), $v_{\text{inflow}}$ 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

$$\sin \gamma(H) = \frac{r}{\sqrt{r^2 + L_f^2(H)}},$$

(7.24)

We define

$$\beta(H) := \frac{r^2}{r^2 + L_f^2(H)}.$$  

(7.25)

Using (7.20)–(7.25), we find

$$\frac{\beta(H) - \beta(H/2)}{\beta(H/2) - \beta(H/4)} = \frac{\sin^2 \gamma(H) - \sin^2 \gamma(H/2)}{\sin^2 \gamma(H/2) - \sin^2 \gamma(H/4)}$$

$$= \frac{\nu_{\text{lam}}^2(H) - \nu_{\text{lam}}^2(H/2)}{\nu_{\text{lam}}^2(H/2) - \nu_{\text{lam}}^2(H/4)} \approx \frac{D_{\text{art}}(H) - D_{\text{art}}(H/2)}{D_{\text{art}}(H/2) - D_{\text{art}}(H/4)} \lesssim 2.$$  

(7.26)

To verify the asymptotic behavior predicted by (7.26), we have to evaluate the quotient at the left hand side. For this, we need to know the radius $r$ of the base of the flame, see
Numerical simulation of the Bunsen flame

Table 7.4: Numerical verification of the relation between the grid spacing and the flame length.

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

(7.25), which is unknown. We estimate $r$ by the radius $r_1$ of the inner jet, see Figure 2.1, so that $r = 0.50$ cm. Using the flame lengths found in our LDC simulations as well as the flame length on the coarsest tensor-product grid, cf. Table 7.2, we can calculate three quotients; they are listed in Table 7.4. The quotients found agree reasonably well with the value of 2 as predicted by (7.26).

Note that we expect from our analysis that the increase of the flame length is strongest when the first level of refinement is added. The flame length still increases, but less rapidly, with additional refinement. This holds true for the first three values of $\Delta L_f$, which are 0.1 cm, 0.06 cm, and 0.04 cm, respectively. The next value of $\Delta L_f$ is again 0.04 cm though and breaks the pattern. This is reflected in the final ratio value (1.0984) in Table 7.4 which deviates more from 2 than the other two ratios listed.

Although simulation results on finer grids are not available, (7.26) enables us to predict the flame length on grids with even higher resolutions. Carrying out the extrapolation on the available data and assuming that (7.26) holds, we find that the flame length would increase further, though less rapidly. It would slowly converge to a limit value of $L_f = 0.76$ cm.
7.3 Numerical results

Figure 7.3: Results for the Bunsen flame calculated with the LDC method with two additional levels of refinement.
Figure 7.4: Plots of the methane mass fraction on the finest level for the LDC experiments.
Chapter 8

Conclusions and recommendations

In this thesis we have studied the local defect correction (LDC) method. LDC is an iterative method for solving elliptic boundary value problems on composite grids based on using simple data structures and simple discretization stencils on uniform or tensor-product grids. Fast solution techniques exist for solving the system of equations resulting from discretization on a structured grid. The discretization on the composite grid is implicitly given by the LDC iteration. Numerical experiments illustrate the fast convergence of the method. The standard method has been combined with multi-level adaptive gridding and domain decomposition. The domain decomposition algorithm provides a natural way for parallelization and enables the usage of many small tensor-product grids rather than a single large unstructured grid. It has been shown that this may greatly reduce memory usage.

The properties above have been illustrated by successfully applying the adaptive multi-level LDC algorithm with domain decomposition to a combustion problem. The mathematical model is a system of nonlinear partial differential equations with strongly nonlinear chemical source terms. The solutions of the system have large gradients in a very small part of the domain and are smooth elsewhere. The simulation results have been compared to those found in literature, and indeed agree well.

We have generalized the standard LDC method to finite volume discretizations. We have shown that the new LDC method with a finite volume adapted correction term has the following attractive features. Like the standard method, it gives a discretization on locally refined grids, and the method is simple: it only uses standard finite volume discretizations on uniform (global coarse and local fine) grids. Most importantly, a discrete conservation property holds for the discretization on the composite grid. The conservation property does not hold for a straightforward generalization of the standard LDC method to finite volume discretizations. In numerical experiments, we have shown that
the new method converges fast (a property it shares with the original LDC method), and we have illustrated the importance that the discrete conservation property may have.

For a model problem, we have derived an upper bound for the norm of the iteration matrix \(M\) of the (standard) LDC algorithm. We managed to show that \(\|M\|_\infty \leq C H^2\), in which \(C\) is a constant and \(H\) is the grid size of the global coarse grid. Even though the proof was given for the relatively simple problem of Poisson's equation on the unit square with Dirichlet boundary conditions, we consider it to be an important step in understanding the convergence behavior of the LDC method. Future research may generalize the proof to a wider class of problems. First steps have already been taken to extend the proof to convection-diffusion equations with constant linear convection. Another interesting research topic is the relation between the width of the safety region and the norm of the iteration matrix. The current result states that \(\|M\|_\infty = O(H^2)\) if we take the width \(\epsilon\) of the safety region positive and independent of \(H\). It would be interesting to know the asymptotic behavior of \(\|M\|_\infty\) if we take \(\epsilon = k H\), \(k\) integer. In numerical experiments we have seen that \(\|M\|_\infty \neq O(H^2)\) in these cases. The convergence is however still very fast, even when \(\epsilon = 0\). For this reason, usage of a safety region does not seem to be necessary in practice.

The adaptive multi-level LDC method with domain decomposition presented in Chapter 6 of this thesis should be considered a first step towards using the LDC method in a complete adaptive grid refinement algorithm. For a given solver for tensor-product grids, a framework has been set up to find an approximation on a locally refined composite grid. Many of the building blocks in this framework are not directly related to the specific application problem under consideration. Parts of the algorithm such as the calculation of a weight function, the flagging of boxes for refinement, the domain decomposition algorithm, and the (composite grid) interpolation routines are problem-independent. The problem-specific solver is used as a black box solver in the LDC framework to find approximations on tensor-product grids only.

The flame simulations performed with the LDC method have shown one drawback of the currently used domain decomposition algorithm, namely its slow convergence. Future research should be directed at using more advanced domain decomposition methods, or, alternatively, at the option of lifting the constraint that all fine grids should cover rectangular areas. At the price of a slightly more difficult data structure, one could use grids in nonrectangular areas, and still employ LDC for the coupling between grids at different refinement levels. Other directions of future work may include sophistication of individual blocks of the LDC algorithm, such as the inclusion of higher order derivatives in the weight function for flagging boxes. In our experiments we have used refinement factors \(\sigma\) equal to 2 and 3. However, LDC allows for values of \(\sigma\) much larger than 2, so that one may add a single level of refinement with a larger value of \(\sigma\) rather than two or more levels with \(\sigma\) small. Further possibilities are the application of the LDC method to a wider class of problems, including time-dependent problems, and usage of artificial boundary conditions of a type other than Dirichlet at the interfaces between coarse and fine grids.
Bibliography


Index

activation energy, 13
adaptive grid refinement, 7 75 82
adaptivity, 7 75
AMR, 87–88
Arrhenius’ law, 13
axis of symmetry, 19 97

Bi-CGSTAB, 8 98
blow-off, 4 19
boundary condition
  artificial, 24 38 75
  for Bunsen flame, 18
box, 75 77
  high activity, 75
Bunsen burner, 1
Bunsen flame, 1 17 20 95 106

calculation history, 86
cell-centered, 34 44
central difference discretization, 22
characteristic function, 28
coarse grid correction step, 27
combustion equation, 1 11 15
composite grid, see grid, composite
composite equation, 11 15
continuity equation, 12
convergence factor, 70
cylindrical coordinates, 15
damping, 78 97
defect, 5 27 38
  local, 27
Δ, 24
density, 1 11
diffusion coefficient, 12
diffusion velocity, 12
domain decomposition, 7 75 84 88

energy, 1 13
energy equation, 13
enthalpy, 1 14
equation of state, 14 18
equidistribution, 78
equivalence ratio, 18
expansion equation, 4

FAC, 7
Fick’s law, 12
fixed point, 5 29 49 50
flagging, 78 79
flame length, 8 102
flash-back, 4 19
flow equation, 1 11 15
flow velocity, 12
  mass-weighted average, 12
flux, 38
frequency factor, 13
FVE, 7

gas constant, 13
global coarse grid, see grid, global coarse
gravitational acceleration, 13 16
grid
  composite, 5 24 36 37
  global coarse, 5 21 24 76
  local fine, 5 21 24 79
  grid function, 24

heat at constant pressure, 14 15
heat flux, 13
heat of formation, 14
heat release, 18
hydrostatic pressure, 13

inflow boundary, 19
integral formulation, 34
<table>
<thead>
<tr>
<th>Term</th>
<th>Page(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>interface</td>
<td>6, 24, 25, 37</td>
</tr>
<tr>
<td>internal energy</td>
<td>13</td>
</tr>
<tr>
<td>interpolation</td>
<td></td>
</tr>
<tr>
<td>linear</td>
<td>24, 32, 61, 73</td>
</tr>
<tr>
<td>quadratic</td>
<td>32, 61</td>
</tr>
<tr>
<td>trigonometric</td>
<td>52, 54, 61, 73</td>
</tr>
<tr>
<td>interpolation operator</td>
<td>24</td>
</tr>
<tr>
<td>Jacobi iteration</td>
<td>78</td>
</tr>
<tr>
<td>Jacobi matrix</td>
<td>98</td>
</tr>
<tr>
<td>Laplacian operator</td>
<td>24</td>
</tr>
<tr>
<td>layering</td>
<td>78, 80</td>
</tr>
<tr>
<td>layering parameter</td>
<td>80</td>
</tr>
<tr>
<td>LDC, 5, 21, 32, 36, 44</td>
<td></td>
</tr>
<tr>
<td>LDC algorithm</td>
<td></td>
</tr>
<tr>
<td>multi-level</td>
<td>81</td>
</tr>
<tr>
<td>multi-level, with domain decomposition</td>
<td>85</td>
</tr>
<tr>
<td>multi-level, with domain decomposition</td>
<td>93</td>
</tr>
<tr>
<td>and regridding</td>
<td></td>
</tr>
<tr>
<td>two-grid, with a priori chosen area of</td>
<td>28</td>
</tr>
<tr>
<td>refinement</td>
<td></td>
</tr>
<tr>
<td>two-grid, with adaptive gridding</td>
<td>79</td>
</tr>
<tr>
<td>with finite volume adapted correction</td>
<td>40</td>
</tr>
<tr>
<td>term</td>
<td></td>
</tr>
<tr>
<td>Lewis number</td>
<td>16</td>
</tr>
<tr>
<td>local defect, see defect, local</td>
<td></td>
</tr>
<tr>
<td>local defect correction, see LDC</td>
<td></td>
</tr>
<tr>
<td>local discretization error</td>
<td>22, 27</td>
</tr>
<tr>
<td>local fine grid, see grid, local fine</td>
<td></td>
</tr>
<tr>
<td>local rectangular refinement, see LRR</td>
<td></td>
</tr>
<tr>
<td>local uniform grid refinement, 5, 56, 56</td>
<td></td>
</tr>
<tr>
<td>LRR, 8, 17, 100, 101</td>
<td></td>
</tr>
<tr>
<td>Mach number</td>
<td>3</td>
</tr>
<tr>
<td>mass density</td>
<td>11</td>
</tr>
<tr>
<td>mass fraction</td>
<td>11</td>
</tr>
<tr>
<td>maximum level of refinement</td>
<td>81</td>
</tr>
<tr>
<td>molar weight</td>
<td>13</td>
</tr>
<tr>
<td>multi-level refinement</td>
<td>7, 75, 82</td>
</tr>
<tr>
<td>Newton's method</td>
<td>97</td>
</tr>
<tr>
<td>Newtonian fluid</td>
<td>13</td>
</tr>
<tr>
<td>one-step chemistry</td>
<td>4, 17</td>
</tr>
<tr>
<td>outflow boundary</td>
<td>20</td>
</tr>
<tr>
<td>overlap</td>
<td>8, 84</td>
</tr>
<tr>
<td>partial differential equation</td>
<td></td>
</tr>
<tr>
<td>elliptic</td>
<td>5, 7, 16, 23, 75</td>
</tr>
<tr>
<td>hyperbolic</td>
<td>5, 87</td>
</tr>
<tr>
<td>parabolic</td>
<td>5</td>
</tr>
<tr>
<td>partitioning</td>
<td>26</td>
</tr>
<tr>
<td>Poisson's equation</td>
<td>7, 30, 52</td>
</tr>
<tr>
<td>Prandtl number</td>
<td>18</td>
</tr>
<tr>
<td>pressure</td>
<td>13, 18</td>
</tr>
<tr>
<td>pressure correction scheme</td>
<td>4</td>
</tr>
<tr>
<td>pseudo time-stepping</td>
<td>98</td>
</tr>
<tr>
<td>quasi-uniformity</td>
<td>77</td>
</tr>
<tr>
<td>quenching</td>
<td>4</td>
</tr>
<tr>
<td>rate constant</td>
<td>13</td>
</tr>
<tr>
<td>reaction rate</td>
<td>11, 12</td>
</tr>
<tr>
<td>refinement factor</td>
<td>37, 79, 108</td>
</tr>
<tr>
<td>regridding</td>
<td>7, 23, 25, 92, 94</td>
</tr>
<tr>
<td>restriction operator</td>
<td>27</td>
</tr>
<tr>
<td>Richardson extrapolation</td>
<td></td>
</tr>
<tr>
<td>safety region</td>
<td>27, 79, 84</td>
</tr>
<tr>
<td>Schwarz procedure</td>
<td>84</td>
</tr>
<tr>
<td>slit burner</td>
<td>1</td>
</tr>
<tr>
<td>smoothing filter</td>
<td>77</td>
</tr>
<tr>
<td>stoichiometric coefficient</td>
<td>12</td>
</tr>
<tr>
<td>stress tensor</td>
<td>13</td>
</tr>
<tr>
<td>subequidistribution</td>
<td>77</td>
</tr>
<tr>
<td>temperature</td>
<td>11, 13</td>
</tr>
<tr>
<td>tensor-product grid</td>
<td>75</td>
</tr>
<tr>
<td>thermal conductivity</td>
<td>14</td>
</tr>
<tr>
<td>thermodynamic identity</td>
<td>14</td>
</tr>
<tr>
<td>tree representation</td>
<td>86</td>
</tr>
<tr>
<td>universal gas constant</td>
<td>13</td>
</tr>
<tr>
<td>upwind discretization</td>
<td>96</td>
</tr>
<tr>
<td>velocity</td>
<td>11, 12</td>
</tr>
<tr>
<td>vertex-centered</td>
<td>34, 44</td>
</tr>
<tr>
<td>viscosity coefficient</td>
<td>13</td>
</tr>
<tr>
<td>viscous stress tensor</td>
<td>13</td>
</tr>
<tr>
<td>vorticity</td>
<td>16</td>
</tr>
<tr>
<td>vorticity-velocity formulation</td>
<td>16</td>
</tr>
<tr>
<td>weight function</td>
<td>7, 75, 77, 78</td>
</tr>
</tbody>
</table>
Summary

Combustion processes are of fundamental importance both for industry and for ordinary life. Numerical simulations may be used as a design tool for the development of more efficient burners with a lower exhaust of polluting gases. In the mathematical description of a flame, we consider it a flowing gas mixture in which chemical reactions take place. The mathematical model follows from the conservation laws for mass and momentum (the flow equations) and the conservation laws for the mass fractions of the species in the mixture and for energy (the combustion equations).

The system of equations describing a flame has several characteristics that make numerical flame simulations a challenge even with present day computer technology. The system is a set of nonlinear partial differential equations, which may describe complex flows. The chemical source terms are strongly nonlinear. Detailed chemical models contain many reactions and many chemical species. Reactions may occur with very different time scales. Finally, there are large differences in geometric scales: the dependent variables have large gradients in the flame zone and are relatively smooth elsewhere.

In this thesis, we focus on the latter problem and study a method for adaptive grid refinement. Rather than using a truly nonuniform grid, we present a method called local defect correction (LDC) that is based on local uniform grid refinement. Advantages of the LDC method include the usage of simple data structures and simple accurate discretization stencils. In the LDC method, the discretization on the composite grid is based on a combination of standard discretizations on several uniform grids with different grid sizes that cover different parts of the domain. At least one grid, the coarse grid, should cover the entire domain, and its grid size should be chosen in agreement with the relatively smooth behavior of the solution outside the high activity areas. Apart from this global coarse grid, one or several local fine grids are used which are also uniform. Each of the local grids covers only a (small) part of the domain and contains a high activity region. The grid sizes of the local grids are chosen in agreement with the behavior of the continuous solution in that part of the domain.

The LDC method 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 right hand side of the coarse grid problem. At each iteration step, the process yields a discrete approximation of the continu-
The LDC method was originally introduced in combination with finite difference discretizations. In a straightforward generalization of the LDC algorithm to finite volume discretizations, the discrete conservation property, which is one of the main attractive features of a finite volume method, does not hold for the composite grid approximation. We present a modified LDC method, which is based on a special form of the defect correction term used in the right hand side of the coarse grid problem. Due to this finite volume adapted defect correction term, the conservation property is preserved in the discretization on the composite grid.

Several properties hold for the fixed point of the LDC iteration. Therefore, it is important to study the convergence behavior of the algorithm. We derive an upper bound for the norm of the iteration matrix for the model problem of Poisson’s equation on the unit square with Dirichlet boundary conditions. If we use trigonometric interpolation on the interface between the fine and coarse grids, we can derive an upper bound for the (infinity) norm of the iteration matrix $M$ of the form $\|M\|_\infty \leq CH^2$, in which $C$ is a constant and $H$ is the grid size of the global coarse grid. Numerical experiments show that the asymptotic behavior of $\|M\|_\infty$ is indeed as predicted by the theoretical results.

The LDC technique is extended by adding adaptivity, multi-level refinement, domain decomposition and regridding to the standard algorithm. The local fine grid is no longer a priori chosen: based on a weight function that measures the smoothness of the solution of the partial differential equation under consideration, high activity areas are determined and flagged for refinement. The flagged boxes are covered with a rectangular patch. In the patch, a local fine grid is chosen. The solution procedure may be applied recursively, i.e., the rectangular patches used to cover high activity areas in the coarse grid may be refined themselves. The usage of a single rectangular patch to cover all flagged boxes in a grid may cause refinement of a large number of unflagged boxes. To remedy this inefficiency we combine the adaptive multi-level LDC algorithm with domain decomposition. Finally, we note that refining a grid and solving the boundary value problem on the new composite grid may cause the area of high activity to move. When this happens, areas of the grid may be refined unnecessarily or areas may not be refined whereas they do need refinement. Therefore, we formulate a regridding procedure.

We apply our proposed adaptive multi-level LDC algorithm with domain decomposition to a combustion problem, namely the simulation of an axisymmetric laminar Bunsen flame. We use a simple one-step chemistry model. We outline the discretization of the system of partial differential equations with the finite difference method and sketch the solution process used on the individual tensor-product grids. Our simulation results show that all dependent variables except for the nitrogen mass fraction have large gradients in the flame zone. A remarkable characteristic of the Bunsen flame problem is that the size of the flame increases on the finer grids. The structure of the flame is similar on all grids. We verify the increase of the flame length using Richardson extrapolation. The numerical results are compared to those found in literature.
Samenvatting

Verbrandingsprocessen zijn zeer belangrijk zowel in de industrie als in huishoudens. Bij huishoudelijke toepassingen kan men denken aan apparaten als boilers en cv-ketels. Computersimulaties kunnen een hulpmiddel zijn bij het ontwerpen van zuinige branders die weinig schadelijke stoffen uitstoten. Om een vlam wiskundig te beschrijven, beschouwen we deze als een stromend gasmengsel waarin chemische reacties plaatsvinden. Het model volgt uit de wetten van massa- en impulsbehoud (de stromingsvergelijkingen) en uit de behoudswetten voor de maassafracties van de stoffen in het mengsel en de wet van energiebehoud (de verbrandingsvergelijkingen).

Het systeem van vergelijkingen dat een vlam beschrijft heeft bepaalde eigenschappen die numerieke simulaties van vlammen zeer kostbaar maken, zelfs op moderne computers. Het model bestaat uit een stelsel niet-lineaire partiële differentiaalvergelijkingen, die ingewikkelde stromingen kunnen beschrijven. De chemische brontermen zijn sterk niet-lineair. Gedetailleerde chemische modellen bestaan uit vele reacties met een groot aantal stoffen. De reacties kunnen sterk uiteenlopende tijdschalen hebben. Ten slotte zijn er grote verschillen in de ruimtelijke schalen in een vlam: de afhankelijke variabelen zoals de temperatuur en maassafracties veranderen sterk van waarde in de voorverwarmings- en reactiezone en kennen een glad verloop buiten deze gebieden.

In dit proefschrift concentreren we ons op het probleem van de uiteenlopende ruimtelijke schalen in een vlam. We bestuderen een methode, lokale defect correctie (LDC) geheten, die is gebaseerd op lokale, uniforme roosterverfijning. Er zijn diverse voordelen van deze aanpak. Allereerst leidt het gebruik van lokale, uniforme roosters tot eenvoudige datastructuren en discretisaties. Verder bestaan er snelle oplosmethoden voor de systemen die men vindt bij discretisatie op een uniform rooster. De discretisatie op het samengestelde rooster volgt uit standaarddiscretisaties op een aantal uniforme roosters met verschillende maaswijdtes die verschillende delen van het domein bedekken. Ten minste één rooster, het grove rooster, overdekt het hele rekendomein. De maaswijdte van het grove rooster moet overeenstemmen met het relatief gladde verloop van de oplossing buiten de gebieden van hoge activiteit. Naast het globale, grove rooster worden een of meer lokale, fijne roosters gebruikt. Deze roosters hebben een maaswijdte die overeenstemt met het gedrag van de continue oplossing in het gebied dat ze overdekken.

De LDC-methode is een iteratief proces: de discretisatie op het grove rooster wordt ver-
Samenvatting

beterd met behulp van lokale discretisaties in subgebieden. De grofroosteroplossing wordt verbeterd door een defect correctie term in het rechterlid van de grofroosterdiscretisatie te plaatsen. Iedere iteratie produceert een discrete benadering van de continue oplossing op het samengestelde rooster; het discrete probleem dat feitelijk wordt opgelost volgt impliciet uit het iteratieve proces.

In zijn oorspronkelijke vorm is het LDC-algoritme geformuleerd in termen van eindige differentie discretisaties. Bij een directe generalisatie naar eindige volume discretisaties gaat de discrete behoudseigenschap verloren voor de discretisatie op het samengestelde rooster. Discreet behoud is een van de aantrekkelijke eigenschappen van een eindige volume methode. We formuleren een nieuwe LDC-methode die is gebaseerd op een speciale defect correctie term voor de grofroosterdiscretisatie. Dankzij deze speciaal voor eindige volumen aangepaste correctie term geldt de behoudseigenschap ook op het samengestelde rooster.

Men kan laten zien dat het vaste punt van de LDC-iteratie een aantal prettige eigenschappen bezit. Een voorbeeld hiervan is de behoudseigenschap voor de combinatie van LDC met eindige volume discretisaties. Het is derhalve belangrijk de convergentie-eigenschappen van de LDC-methode nader te onderzoeken. Voor een modelprobleem, namelijk de Poissonvergelijking op het eenheidsvierkant met Dirichlet randvoorwaarden, geven we een bovengrens voor de norm van de iteratiematrix. Als we voor de interpolatie op de interface tussen het grove en het fijne gebied trigonometrische interpolatie gebruiken, dan kunnen we laten zien dat de norm van de iteratiematrix $M$ voldoet aan $\|M\|_\infty \leq CH^2$. Hierbij is $C$ een constante en is $H$ de maaswijdte van het grove, globale rooster. Numerieke experimenten laten zien dat het door de theorie voorspelde asymptotisch gedrag inderdaad optreedt.

We breiden de LDC-methode uit met adaptieve, multi-level roosterverfijning. Het lokale, fijne rooster wordt nu niet tevoren vastgelegd. Op grond van een functie die de gladheid van de oplossing van het randwaardenprobleem weerspiegelt, worden gebieden met hoge activiteit geïdentificeerd en gemarkend voor verfijning. De gemarkande cellen in het rooster worden overdekt met een rechthoek waarin een lokaal, fijn rooster wordt gekozen. Dit principe kan recursief worden toegepast. Bij het overdekken van de gemarkande cellen met slechts één rechthoek kan het gebeuren dat niet-gemarkeerde cellen ook worden verfijnd. Dit is ongewenst en om het verfijnen van niet-gemarkeerde cellen te beperken, combineren we de LDC-methode met domeindecompositie.

We passen het adaptieve, multi-level LDC-algoritme met domeindecompositie toe op een verbrandingsprobleem, namelijk de numerieke simulatie van een axisymmetrische laminaire Bunsenvlam. Voor de chemische reacties hanteren we een eenstapsmodel. We geven de gebruikte (eindige differentie) discretisatie van de partiële differentiaalvergelijkingen en schetsen de oplosmethode die wordt gebruikt op de afzonderlijke tensor-product roosters. Onze simulaties laten zien dat alle variabelen (behalve de massafractie van stikstof) een steil verloop kennen in het gebied waar zich de vlam bevindt. Een opvallende eigenschap van de Bunsenvlam is de toename van de lengte van de vlam op fijner roosters. De structuur van de vlam is soortgelijk op alle roosters. De gevonden resultaten worden vergeleken met die in de literatuur.
Curriculum vitae

The author of this thesis was born May 12th, 1972, in Breda, the Netherlands. He finished his preuniversity education at the Stedelijk Gymnasium Breda in 1990. Martijn started studying both mathematics and computer science at Eindhoven University of Technology, and received the propaedeutic diplomas for both studies a year later. At this point, he specialized in mathematics. During his studies, Martijn helped to organize a study tour to Moscow and St. Petersburg for the local study society of mathematics and computer science students. In 1995, Martijn graduated with honors. His master's thesis, written under the supervision of prof.dr.ir. J. de Graaf, was titled Plane Stokes flow with a free boundary driven solely by surface tension.

After his studies at Eindhoven University, Martijn was selected for the Japan Prizewinners Program, a one-year postgraduate course financed by the Dutch Ministry of Education. For four months, Martijn participated in Japanese language training and lecture series at Leiden University. From January till August 1996, Martijn lived in Tokyo, Japan, and worked at the External Affairs Department of the Hitachi headquarters. During this period he attended further courses at the Japan-Netherlands Institute in Tokyo.

Late 1996, Martijn returned to the Netherlands, and in January 1997, he started working as a PhD student in the Scientific Computing Group of prof.dr. R.M.M. Mattheij at Eindhoven University of Technology. His research on local defect correction techniques has lead to this thesis. During his PhD research, Martijn made a three-month research visit to Yale University, New Haven, CT, where he was a guest in the group of prof.dr. M.D. Smooke. Next to his PhD work, Martijn helped to initiate and organize the PhDays, an informal weekend for Dutch and Flemish PhD students in numerical mathematics. He has also been an active member of both the mathematics alumni society of Eindhoven University and of the alumni association of the aforementioned Japan program.

Since January 1st, 2001, Martijn works as an assistant professor in the Scientific Computing Group at Eindhoven University of Technology.