Multigrid Methods for Obstacle Problems

by

Chunxiao Wu

A Research Paper
presented to the University of Waterloo
in partial fulfillment of the
requirement for the degree of
Master of Mathematics
in
Computational Mathematics

Supervisor: Prof. Justin W.L. Wan

Waterloo, Ontario, Canada, 2013

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Obstacle problems can be posed as elliptic variational inequalities, complementarity inequalities and Hamilton-Jacobi-Bellman (HJB) partial differential equations (PDEs). In this paper, we propose a multigrid algorithm based on the full approximate scheme (FAS) for solving the membrane constrained obstacle problems and the minimal surface obstacle problems in the formulations of HJB equations. A special coarse grid operator is proposed based on the Galerkin operator for the membrane constrained obstacle problem in this paper. Comparing with standard FAS with the direct discretization coarse grid operator, the FAS with the proposed operator converges faster. Due to the nonlinear property of the minimal surface operator, the Galerkin operator for the minimal surface obstacle problem is not accurate. We introduce the direct discretization operator for the minimal surface obstacle problem. A special prolongation operator based on the bilinear interpolation is proposed to interpolate functions from the coarse grid to the fine grid. At the boundary between the active set and inactive set, the proposed prolongation operator can capture the active grid points and put accurate values at these points. We will demonstrate the fast convergence of the proposed multigrid method for solving obstacle problems by comparing with other multigrid methods.

Acknowledgements

I would like to thank my supervisor Prof. Justin W.L. Wan for his help and patience thoughout the year. I also want to thank all of my classmates for their support.

Dedication

This is dedicated to my lovely mom and my sweet dad.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The obstacle problem is a problem that finds the equilibrium position of an elastic membrane with fixed boundary and is constrained to lie below and/or above the given obstacles. The problem can be posed as a partial differential equation (PDE). It is regarded as an example in the study of elliptic variational inequalities and free boundary problems. Many problems in real life such as porous flow through a dam, minimal surface over an obstacle, American options, elastic-plastic cylinder, etc. can be reformulated as elliptic variational inequalities [5][20]. A free boundary problem is a partial differential equation that has to be solved for both an unknown function and an unknown domain [7]. An instance of the free boundary problem is the melting of ice. Given a block of ice, one can solve the heat equation with an appropriate initial guess and given boundary conditions. However, if the temperature of a specific region is greater than the melting point of ice, this domain of ice will be melting and occupied by liquid water. The solution to the PDE will give the dynamic location of the interface between the ice and liquid. We call these obstacle type problems.

Obstacle problems can also be formulated as linear complementarity problems and Hamilton-Jacobi-Bellman (HJB) equations [11]. However, it is generally difficult to compute the exact solutions of these problems, so we will focus on the numerical solutions here in this paper. The way to project the continuous obstacle problem into a discrete problem is through discretization. There are several techniques that may be used to discretize a PDE. For instance, finite difference, finite element and finite volume methods [14]. Finite difference methods are widely used, since they are intuitive and easy to implement. Finite element methods are commonly used with the domain decomposition and subspace correction methods.

1.1 Overview Methods to Solve Obstacle Problems

Many methods are introduced to solve elliptic variational inequality PDEs. Existing approaches include projected relaxation methods, classical stationary iterative methods, conjugate gradient methods, multigrid methods and so on [11][16][17].

The classical stationary iterative methods are easy to implement but the convergence rate is slow when the problem is large. Projected relaxation methods are regarded as standard techniques to solve elliptic variational inequalities [8][9]. They are known to be easy to implement and convergent. However, the drawback of this method is that it highly depends on the choice of the relaxation parameter and has a slow asymptotic convergence rate.

Various forms of preconditioned conjugate gradient (PCG) algorithms for solving the nonlinear variational inequalities are presented in [16]. The conjugate gradient (CG) method is an iterative method designed to solve certain systems of linear or nonlinear equations. PCG is introduced to accelerate convergence by adding special parameters. It is more efficient than the projected relaxation method. However, the convergence rate still depends on the size of the problem. When the grid size is refined, a PCG method may not be very efficient.

A multigrid method is introduced in [11] to solve the finite difference discretized PDE in the form of the minimal surface obstacle problem. Two phases are introduced in the paper. The aim of the first phase, in which the computations are not expensive, is to get a good initial guess for the iterative phase two. In the second phase, the problem is solved by a W-cycle multigrid method. A cutting function is applied after the coarse grid correction. The convergence is proven to be better than PCG. However, the application of the cutting function and the phase one leads to expensive computations overall.

The elliptic variational inequalities can be reformulated as linear complementarity problems. It is easy to write the linear complementarity problems into PDEs. A multigrid method, namely, projected full approximate scheme (PFAS), is proposed in [3] to solve linear complementarity problems arising from free boundary problems. The proposed multigrid method is based on the full approximate scheme (FAS), which is often used for solving nonlinear PDEs. The multigrid method is built on a generalization of the projected SOR. Two further algorithms established on PFAS are introduced: PFASMD and PFMG, both of which are faster than PFAS. These methods are better than the method in [11] in the sense of the convergence rate.

The multigrid PFAS to solve the American style option problem in the linear complementarity form is introduced in [17], where a Fourier analysis of the smoother is provided.

The American style option problem is an obstacle type problem which can be formulated into PDEs. The F-cycle multigrid method is applied and the comparison between the F-cycle and the V-cycle for solving linear complementarity problems is shown. The F-cycle is faster than the V-cycle as mentioned in [17]. However, the F-cycle requires more computations in each iteration. The additional steps are recombination operations on the fine grid, so it is relatively expensive.

The elliptic variational inequalities can also be reformulated as Hamilton-Jacobi-Bellman (HJB) equations. A multigrid algorithm which involves an outer and an inner loop is developed in [12]. The active and inactive sets of all grid levels are computed and stored in the outer loop. The W-cycle FAS multigrid method is applied to solve a linear PDE in the inner loop which is nested in the outer loop. An iterative step is adopted to give a good initial guess starting from the biggest grid size up to the fine grid. Both [11] and [12] are trying to use the active and inactive sets strategy to separate the obstacles from the solving of the direct PDEs. The computational complexity largely depends on the stopping criterion for the inner loop. If the stopping criterion is not chosen wisely, the computations can be very expensive.

A mutilevel domain decomposition and subspace correction algorithm is applied in [21] to solve the HJB equations for obstacle problems over a minimal surface. A special interpolator is chosen to solve the obstacle problem over a convex space. The proof of the linear rate of convergence for proposed algorithms is provided.

A so-called monotone multigrid method and a truncated version is introduced in [13] to solve the obstacle problem. The convergence rate of the proposed method for solving the minimal surface obstacle problem is proved as the same efficiency as for the classical multigrid for the unconstrained case. The finite element method is used for discretization.

1.2 Objective

We will consider two kinds of obstacle problems. They share the same boundary conditions and obstacle constraints, but with different partial differential operators. One is with the Laplacian operator, and the other operator we call it the minimal surface operator. The obstacle problems can be reformulated as HJB equations. The equations are discretized by a finite difference method. This research paper is to develop an efficient multigrid method for solving the discrete HJB equations. The main results of this research paper are:

- The FAS multigrid method with the Galerkin coarse grid operator is introduced and applied on the membrane constrained obstacle problem.

- The FAS multigrid method with the direct discretization operator is presented and used to solve the minimal surface obstacle problem.
- The performances of the FAS method for solving both obstacle problems are analyzed.

This research paper is arranged as follows: the obstacle problems, their mathematical derivations and their discretization are given in Chapter 2. Chapter 3 presents the multigrid methods, in which the V-cycle is shown and the corresponding algorithm is given. A linear problem is considered as an example to introduce multigrid methods. The FAS V-cycle to solve obstacle problems are presented in Chapter 4, where the algorithms for the two kinds of obstacle problems are introduced. Numerical experiments are discussed in Chapter 5 and we summarize the work of this research paper in Chpater 6.

Chapter 2

Obstacle Problem

The obstacle problem discussed in this research paper is originally introduced for finding the equilibrium position of a membrane above a constraint of an obstacle, with fixed boundary. In classical linear elasticity theory, the obstacle problem is regarded as one of the simplest unilateral problems, and also as geometrical problems [20]. The obstacle problems can be modeled as elliptic variational inequalities and free boundary problems [5]. Many variational inequalities problem, for example, the elastic-plastic torsion problem and the cylindric bar, are regarded as obstacle type problems by means of the membrane analogy. Another example is a cavitation problem in the theory of lubrication.

The solution to the obstacle problem can be separated into two different regions. One region is where the solution equals the obstacle value, also known as the active set, and the other region is with the solution above the obstacle. The interface between the two regions is called the free boundary, so the obstacle problem is also studied as the free boundary problem [5]. Figure 2.1 shows an example of the obstacle (left) and the solution to the obstacle problem (right). In this chapter, we will derive the formulations of two kinds of obstacle problems: membrane constrained and minimal surface obstacle problems.

2.1 Deformation of a Membrane Constrained by an Obstacle

A membrane considered in classical elasticity theory is a thin plate, which has no resistance to bend, but would act in tension [20]. We assume a membrane in a domain $\Omega \subset \mathbb{R}^2$ on the plane Oxy is equally stretched by a uniform tension, and also loaded with the external

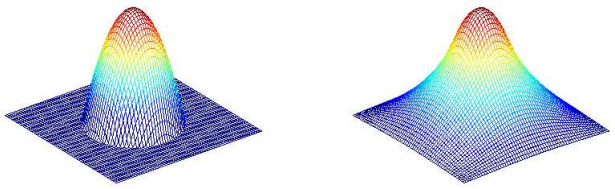


Figure 2.1: An obstacle problem example, (left) the obstacle, (right) the solution

force f, which is also uniformly distributed. We assume each point (x, y) of the membrane is replaced by a numerical amount $u(x, y)$ perpendicular to the domain Ω. The boundary of the domain Ω is represented as $\partial\Omega$. For simplicity, from now on, we use X stands for the two dimensional point (x, y). The membrane problem considered here is defined on a convex domain, with the function $u(X)$ and boundary conditions $u|_{\partial\Omega} = g(X)$ with some obstacle $\psi(X)$ in Ω. To be more precise, we are given
(a) A convex domain Ω,
(b) $u(X) = g(X)$ on the boundary $\partial\Omega$, to be specific, $g(X) = 0$ is adopted in this paper,
(c) An obstacle function $\psi(X)$.

According to the position of the obstacle, we can classify the obstacle problem into three classes: (a) the lower obstacle problem, in which the solution $u(X)$ lies above the obstacle, (b) the upper obstacle problem, in which the solution $u(X)$ lies below the obstacle, and (c) the two sided obstacle problem, in which the solution $u(X)$ lies in between the two sides obstacles. In this chapter, we will take the lower obstacle problem as an example to derive the mathematical formulations.

Here we define a close convex set K to be

$$K = \{u \in V, u|_{\partial\Omega} = g(X), u \geq \psi\}, \tag{2.1}$$

where V stands for the lower obstacle problem space which is the H^1 space considered here.

In the following, we will give three different formulations of the membrane constrained obstacle problem: an elliptic variation inequality, a linear complementarity problem and an HJB equation.

2.2 Elliptic Variational Inequality Formulation of Membrane Constrained Obstacle Problem

The potential energy of the membrane can be expressed as:

$$D(u) = \int_\Omega \frac{1}{2}|\nabla u|^2 dX, \tag{2.2}$$

where $u \in K$, and ∇u stands for the gradient of u. There may be external forces f, the energy of which is given by

$$F(u) = \int_\Omega fu \; dX. \tag{2.3}$$

So (2.2) together with (2.3) give us the total potential energy $E = D - F$, which is

$$E(u) = \int_\Omega \frac{1}{2}|\nabla u|^2 dX - \int_\Omega fu \; dX. \tag{2.4}$$

The membrane constrained problem is to find u such that it minimizes the potential energy,

$$\min_u \quad E(u), \qquad u \in K, \tag{2.5}$$

subject to the obstacle constraint given by the convex set K. In order to find the solution, we need to make sure u satisfies the Euler's equation and (2.5), which turns out to be the Poisson's equation,

$$-\Delta u \equiv -(u_{xx} + u_{yy}) = f. \tag{2.6}$$

Let $\bar{u}$ be the solution to (2.5), that is,

$$E(\bar{u}) \leq E(v), \qquad \forall v \in K. \tag{2.7}$$

Since K is a convex set, $\bar{u} + \lambda(v - \bar{u}) \in K$ for $\lambda \in [0, 1]$. Define $h(\lambda) \equiv E(\bar{u} + \lambda(v - \bar{u}))$. When $\lambda = 0$, $h(\lambda)$ is minimized and hence $h'(0) \geq 0$. It follows that

$$h'(0) = \lim_{\lambda \to 0^+} \frac{1}{\lambda} \{E(\bar{u} + \lambda(v - \bar{u})) - E(\bar{u})\} \geq 0. \tag{2.8}$$

Substituting (2.4) into (2.8), we obtain,

$$h'(0) = \int_\Omega \nabla \bar{u} \nabla (v - \bar{u}) dX - \int_\Omega f(v - \bar{u}) dX \geq 0. \tag{2.9}$$

Thus, if $\bar{u}$ is a solution to (2.5) (or (2.7)), then $\bar{u}$ also satisfies

$$\bar{u} \in K: \qquad \int_\Omega \nabla \bar{u} \nabla (v - \bar{u}) dX \geq \int_\Omega f(v - \bar{u}) dX, \qquad \forall v \in K, \tag{2.10}$$

which is called an elliptic variational inequality.

Another way to see the solution to (2.10) is also the solution to (2.5) (or (2.7)) is that if u satisfies (2.10) then for $\forall v \in K$,

$$\begin{aligned} E(v) =& E(u + v - u) = E(u + (v - u)) \\ =& E(u) + \int_\Omega \nabla u \nabla (v - u) dX - \int_\Omega f(v - u) dX + \frac{1}{2} \int_\Omega |\nabla (v - u)|^2. \end{aligned} \tag{2.11}$$

Since u satisfies (2.10), $\int_\Omega \nabla \bar{u} \nabla (v - \bar{u}) dX - \int_\Omega f(v - \bar{u}) \geq 0$. Hence we have

$$\begin{aligned} E(v) &\geq E(u) + \frac{1}{2} \int_\Omega |\nabla (v - u)|^2 \\ &\geq E(u). \end{aligned} \tag{2.12}$$

It implies that u is also a solution to (2.5) (or (2.7)).

2.3 Linear Complementarity Formulation of Membrane Constrained Obstacle Problem

In order to derive the linear complementarity formula, we define two sets

$$K_1 = \{u \in K : u(X) = \psi(X)\}, \tag{2.13}$$

and

$$K_2 = \{u \in K : u(X) > \psi(X)\}. \tag{2.14}$$

Take an arbitrary $\phi \in K_2$, such that $\phi(X) = 0$ on $\partial\Omega$. There exists $\epsilon_0 > 0$, such that $v = \bar{u} \pm \epsilon\phi \in K$ for $0 < \epsilon \leq \epsilon_0$. Substituting $v = \bar{u} \pm \epsilon\phi$ into (2.10), we can get

$$\int_\Omega \nabla \bar{u} \cdot \nabla (\pm \epsilon \phi) dX - \int_\Omega f(\pm \epsilon \phi) dX \geq 0. \tag{2.15}$$

Rearranging (2.15), we obtain

$$\pm \epsilon \int_\Omega \left(\nabla \bar{u} \nabla \phi - f\phi \right) dX \geq 0. \tag{2.16}$$

Integrating by parts, (2.16) becomes

$$\pm \epsilon \left\{ [\nabla u \phi]_{\partial\Omega} + \int_{\Omega} (-\Delta \bar{u} - f) \phi dX \right\} \geq 0, \tag{2.17}$$

Since $\phi(X) = 0$ on $\partial\Omega$, we have

$$\pm \epsilon \int_{\Omega} (-\Delta \bar{u} - f) \phi \geq 0, \quad \forall \phi \in K_2. \tag{2.18}$$

Since the inequality holds for both $+\epsilon$ and $-\epsilon$, we can conclude that

$$-\Delta \bar{u} = f, \qquad \bar{u}(X) > \psi(X). \tag{2.19}$$

In the other case, $\bar{u}(X)$ is not strictly greater than $\psi(X)$. We let $\phi \in K$, and $\phi \geq 0$. Let $v = u + \phi \in K$. Then (2.10) becomes:

$$\int_{\Omega} \nabla \bar{u} \cdot \nabla \phi dX - \int_{\Omega} f \phi dX \geq 0. \tag{2.20}$$

Following the same algebra from (2.15) to (2.18), we have

$$\int_{\Omega} (-\Delta \bar{u} - f) \phi \geq 0, \tag{2.21}$$

for any $\phi \geq 0$. This implies that

$$-\Delta \bar{u} - f \geq 0, \qquad \bar{u}(X) \geq \psi(X). \tag{2.22}$$

From (2.19) and (2.22), we can conclude that $-\Delta \bar{u} - f \geq 0$ is true almost everywhere except for $\partial\Omega$ in Ω, where the values are given by the Dirichlet boundary conditions. The two cases (2.19) and (2.22) can be written as a complementarity problem:

$$\begin{aligned} \bar{u} - \psi &\geq 0, \\ -\Delta \bar{u} - f &\geq 0, \\ (\bar{u} - \psi)(-\Delta \bar{u} - f) &= 0. \end{aligned} \tag{2.23}$$

2.4 HJB Equation Formulation of Membrane Constrained Obstacle Problem

Equation (2.23) implies that if $\bar{u} > \psi$, or $\bar{u} - \psi > 0$, then $-\Delta\bar{u} - f = 0$. Otherwise, if $\bar{u} - \psi = 0$, then $-\Delta\bar{u} - f \geq 0$. Hence, the linear complementarity formulation (2.23) is equivalent to

$$\min(-\Delta\bar{u} - f, \bar{u} - \psi) = 0, \quad \text{on } \Omega, \tag{2.24}$$

which is called an Hamilton-Jacobi-Bellman (HJB) equation. Here, $\bar{u} \in K$, where K is defined in (2.1).

This formulation is only for the lower obstacle problem. For the upper obstacle problem, the corresponding HJB equation is given by,

$$\max(-\Delta\bar{u} - f, \bar{u} - \psi) = 0, \quad \text{on } \Omega. \tag{2.25}$$

Here, $\bar{u} \in K^u$, where $K^u = \{u \in V | u \geq \psi^2\}$, and V is the H^1 space. For the two sided obstacle problem, we assume $\bar{u} \in K^{two} = \{u \in V | \psi^1 \leq u \leq \psi^2\}$. The HJB equation is given by

$$\max_{1 \leq \mu \leq 4} [-L^\mu \bar{u} - f^\mu] = 0, \tag{2.26}$$

where

$$\begin{aligned} &- L^\mu u = [sgn(u - \psi^\mu)](-\Delta u), \quad f^\mu = [sgn(u - \psi^\mu)]f, && \mu = 1, 2, \\ &- L^\mu u = (-1)^\mu u, \quad f^\mu = (-1)^\mu \psi^{\mu-2}, && \mu = 3, 4. \end{aligned} \tag{2.27}$$

2.5 Minimal Surfaces with Obstacles

In this section, we introduce another form of the obstacle problem, i.e, the minimal surface with obstacles. It is to describe the equilibrium shape for a mass bounded by a surface. We assume that the superficial tension exerts a pressure that is proportional to the mean curvature at each point. The pressure applied is the same in all directions, such that every point on the surface shares the constant mean curvature. The minimal surface problem is of great interest in the mathematical study [20]. Here, we only consider the special case, that is: in the membrane problem, the minimal surfaces can be represented as a function $u = u(X)$ over a smooth subset Ω in $\mathbb{R}^2$. The boundary condition is also given by $g(X) = 0$ on $\partial\Omega$.

We assume that the potential energy of the membrane is proportional to its area, which is given by

$$\int_\Omega \sqrt{1+|\nabla u|^2}dX, \quad u \in K. \tag{2.28}$$

The minimal surface obstacle problem is to find the minimal potential energy of the membrane and can be stated in the form

$$u \in K: \quad \int_\Omega \sqrt{1+|\nabla u|^2}dX \le \int_\Omega \sqrt{1+|\nabla v|^2}dX, \qquad \forall v \in K. \tag{2.29}$$

The minimal surface obstacle problem is related to the membrane constrained obstacle problem. If we take the Talyor's expansion, the integrand of (2.28) can be written as

$$\sqrt{1+|\nabla u|^2} = 1 + \frac{1}{2}|\nabla u|^2 + O(\nabla u^4). \tag{2.30}$$

For small $|\nabla u|$, $O(\nabla u^4)$ can be discarded. The minimization of $1+\frac{1}{2}|\nabla u|^2$ is the same as minimizing $\frac{1}{2}|\nabla u|^2$, since the constant term dose not affect the minimization. By dropping the high order term $O(\nabla u^4)$, the minimal surface problem becomes the membrane constrained obstacle problem in (2.5).

We can also apply the external forces f here, the energy of which is given in (2.3). Similar to the derivation in Section 2.2, (2.29) can be formulated as a variational inequality:

$$u \in K: \quad \int_\Omega \frac{\nabla u \cdot \nabla(v-u)}{\sqrt{1+|\nabla u|^2}}dX - \int_\Omega f(v-u)dX \ge 0, \qquad \forall v \in K. \tag{2.31}$$

With the same reasoning as in (2.19) and (2.22), we conclude that u must satisfy the following equations:

$$\begin{aligned} Au \equiv -\nabla \cdot \left(\frac{\nabla u}{\sqrt{1+|\nabla u|^2}}\right) = f \quad & \text{for } u(X) > \psi(X), \\ Au \equiv -\nabla \cdot \left(\frac{\nabla u}{\sqrt{1+|\nabla u|^2}}\right) \ge f \quad & \text{for } u(X) \ge \psi(X). \end{aligned} \tag{2.32}$$

The HJB equations for the minimal surface problem can be obtained by replacing the Δu and L^μ in (2.24), (2.25) and (2.26). More precisely, the equations for the lower and upper minimal surface obstacle problems are

$$\begin{aligned} \min(A\bar{u} - f, \bar{u} - \psi) = 0, \quad & \text{on } \Omega, \\ \max(A\bar{u} - f, \bar{u} - \psi) = 0, \quad & \text{on } \Omega, \end{aligned} \tag{2.33}$$

respectively. For the two sided minimal surface obstacle problem, the corresponding HJB equation is given by

$$\max_{1\le\mu\le 4}[G^\mu \bar{u} - f^\mu] = 0 \tag{2.34}$$

where

$$\begin{aligned} &G^\mu u = [sgn(u-\psi^\mu)](Au), \quad f^\mu = [sgn(u-\psi^\mu)]f, \quad \mu = 1,2, \\ &G^\mu u = (-1)^\mu u, \quad f^\mu = (-1)^\mu \psi^{\mu-2}, \quad \mu = 3,4. \end{aligned} \tag{2.35}$$

The term Au is defined in (2.32) for all these three HJB equations.

The minimal surface obstacle problem is related to the membrane constrained obstacle problem as mentioned above. However, the minimal surface operator A here in (2.32) and the Laplacian operator Δ are quite different. The Laplacian operator Δ does not depend on the function u. It is a linear operator. But the minimal surface operator A in (2.32) is nonlinear and highly depends on u.

2.6 Finite Difference Discretization of Membrane Constrained Obstacle Problems

It is difficult to solve the obstacle problem in the HJB form analytically, so we will consider the numerical solution based on a finite difference discretization. Finite difference is a numerical method to approximate derivatives of a function by algebraic formulas derived from Taylor's polynomials. We will first introduce some notations. Consider the Poisson's equation

$$-u_{xx} - u_{yy} = f \quad \text{in } \Omega = (0,1)^2. \tag{2.36}$$

Denote the grid points by (x_i, y_i), where $x_i = i\Delta x$, $y_j = j\Delta y$, $0 \le i,j \le N+1$. $\Delta x = \Delta y = h = \frac{1}{N+1}$ is the grid size. Let $u_{i,j}$ be an approximation to $u(x_i, y_j)$. Then a finite difference discretization of (2.36) can be written as

$$-\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} - \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2} = f_{i,j}, \quad 1 \le i,j \le N. \tag{2.37}$$

Thus, we have a set of N^2 linear equations. We can rewrite the linear system by an $N^2 \times N^2$ matrix A_h^M as

$$A_h^M = \frac{1}{h^2}\begin{pmatrix} D & B & 0 & \cdots & \cdots \\ B & D & B & \cdots & \cdots \\ 0 & B & D & B & \cdots \\ \vdots & \vdots & \vdots & \ddots & \cdots \\ 0 & \cdots & \cdots & B & D \end{pmatrix}, \tag{2.38}$$

where D and B are both $N \times N$ matrices defined as,

$$D = \begin{pmatrix} 4 & -1 & 0 & 0 & \cdots \\ -1 & 4 & -1 & \cdots & \cdots \\ 0 & -1 & 4 & -1 & \cdots \\ \vdots & \vdots & \vdots & \ddots & \cdots \\ 0 & \cdots & \cdots & -1 & 4 \end{pmatrix}, \tag{2.39}$$

$$B = \begin{pmatrix} -1 & 0 & 0 & 0 & \cdots \\ 0 & -1 & 0 & \cdots & \cdots \\ 0 & 0 & -1 & 0 & \cdots \\ \vdots & \vdots & \vdots & \ddots & \cdots \\ 0 & \cdots & \cdots & 0 & -1 \end{pmatrix}. \tag{2.40}$$

The superscript "M" for A_h^M here stands for the membrane constrained obstacle problem.

2.7 Finite Difference Discretization of Minimal Surface Obstacle Problem

Similarly, we apply a finite difference method to discretize the operator in (2.32):

$$A_h^{MS} u_{i,j} = AC u_{i,j} + AW u_{i-1,j} + AE u_{i+1,j} + AS u_{i,j-1} + AN u_{i,j+1}. \tag{2.41}$$

The superscript "MS" in A_h^{MS} here stands for the minimal surface obstacle problem. AC, AW, AE, AS, AN can be understood as the weights of the stencil in the center, west,

east, south, north, respectively. They are given by

$$AW = -\frac{1}{h^2}\left(\frac{1}{2\sqrt{(\frac{u_{i,j}-u_{i-1,j}}{h})^2+(\frac{u_{i,j}-u_{i,j-1}}{h})^2+1}} + \frac{1}{2\sqrt{(\frac{u_{i,j}-u_{i-1,j}}{h})^2+(\frac{u_{i-1,j+1}-u_{i-1,j}}{h})^2+1}}\right),$$

$$AE = -\frac{1}{h^2}\left(\frac{1}{2\sqrt{(\frac{u_{i+1,j}-u_{i,j}}{h})^2+(\frac{u_{i+1,j}-u_{i+1,j-1}}{h})^2+1}} + \frac{1}{2\sqrt{(\frac{u_{i+1,j}-u_{i,j}}{h})^2+(\frac{u_{i,j+1}-u_{i,j}}{h})^2+1}}\right),$$

$$AS = -\frac{1}{h^2}\left(\frac{1}{2\sqrt{(\frac{u_{i,j}-u_{i-1,j}}{h})^2+(\frac{u_{i,j}-u_{i,j-1}}{h})^2+1}} + \frac{1}{2\sqrt{(\frac{u_{i+1,j-1}-u_{i,j-1}}{h})^2+(\frac{u_{i,j}-u_{i,j-1}}{h})^2+1}}\right),$$

$$AN = -\frac{1}{h^2}\left(\frac{1}{2\sqrt{(\frac{u_{i+1,j}-u_{i,j}}{h})^2+(\frac{u_{i,j+1}-u_{i,j}}{h})^2+1}} + \frac{1}{2\sqrt{(\frac{u_{i,j+1}-u_{i-1,j+1}}{h})^2+(\frac{u_{i,j+1}-u_{i,j}}{h})^2+1}}\right),$$

and $AC = -(AW + AE + AS + AN) + 1$. A_h^{MS} can also be written as a matrix form, which is not given here.

It would be nice that the discretization of the minimal surface problem is monotone so that the numerical solution will converge to the viscosity solution. However, it is not obvious one can prove that the given finite difference discretization is actually a monotone scheme. Perhaps it is possible to apply forward or backward differencing to the discretization of the ∇u term in such a way that the resulting discretization method is monotone.

However, it is not clear how it can be done and we will just focus on the given discretization when we apply our multigrid method

2.8 HJB Equation of Discretized Obstacle Problems

Now we consider the discretization of the HJB equations for both obstacle problems, the membrane constrained obstacle problem and the minimal surface obstacle problem.

We give several notations here we will use in the following. We order the grid points $(X_1, X_2, X_3, ..., X_{N^2})$ lexicographically and denote u_h, f_h, as a vector in the same lexicographical order. Thus $A_h(u_h)$ is a vector, and $A_h(u_h)_i$ represents the i^{th} item of the vector.

An i^{th} grid point is called an inactive point if

$$A_h(u_h)_i - f_{h,i} < u_{h,i} - \psi_{h,i}.$$

Otherwise it is called an active point. All the active points build up an active set and the inactive points form an inactive set.

Depending on the type of constraints, we divide the obstacle problem into three types: (1) the lower obstacle, the set $K_h^L = \{v_h \in \mathbb{R}^{N_h} | v_h \geq \psi_h^1\}$, (2) the upper obstacle, the set $K_h^U = \{v_h \in \mathbb{R}^{N_h} | v_h \leq \psi_h^2\}$, (3) the two sided obstacle, $K_h^{two} = \{v_h \in \mathbb{R}^{N_h} | \psi_h^1 \leq v_h \leq \psi_h^2\}$. The HJB equations are given as

$$\min[A_h(u_h) - f_h, u_h - \psi_h^1] = 0, \tag{2.42}$$

for the lower obstacle $K_h = K_h^L$,

$$\max[A_h(u_h) - f_h, u_h - \psi_h^2] = 0, \tag{2.43}$$

for the upper obstacle $K_h = K_h^U$, and

$$\max_{1 \leq \mu \leq 4} [A_h^\mu(u_h) - f_h^\mu] = 0, \tag{2.44}$$

for the two sided obstacle problem $K_h = K_h^{two}$, where

$$\begin{aligned} &A_h^\mu(u_h) = [sgn(u_h - \psi_h^\mu)] A_h(u_h), \quad f_h^\mu = [sgn(u_h - \psi_h^\mu)] f_h, \quad \mu = 1, 2, \\ &A_h^\mu(u_h) = (-1)^\mu u_h, \quad f_h^\mu = (-1)^\mu \psi_h^{\mu-2}, \quad \mu = 3, 4. \end{aligned} \tag{2.45}$$

Here $A_h = A_h^M$, defined in (2.38), for the membrane constrained obstacle problem and $A_h = A_h^{MS}$, given by (2.41), for the minimal surface obstacle problem. Note that the HJB equations should be understood componentwise.

In this research paper, we are going to solve the discrete HJB equation with lower obstacles, i.e (2.42). The discrete HJB equation is difficult to solve numerically because of its nonlinearity. Any grid point, say the i^{th} grid point, is active or inactive depending on both $A_h(u_h)_i - f_{h,i}$ and $u_{h,i} - \psi_{h,i}$. However, we do not know that in advance. It depends on the approximate solution u_h and the obstacle function ψ_h. We need to update the active and inactive set when we update the approximate solution u_h. The active and inactive sets make this problem nonlinear and difficult to solve numerically. For the minimal surface obstacle problem, the operator A_h^{MS} itself is highly nonlinear. So the HJB equation for the minimal surface obstacle problem is even more difficult so solve.

One way to solve the HJB equation is the policy iteration [1]. The policy iteration consists of two loops, i.e. the outer loop and the inner loop. In the outer loop, one can fix the active set and inactive set. In the inner loop, some iterative methods can be applied to solve the linear problem. A W-cycle multigrid method can be taken as an inner loop solution [12]. In practice, the policy iteration converges fast. However, the computation in the inner loop can be costly. So the total computational work may still be expensive.

Chapter 3

Multigrid Methods

This chapter provides several important and basic concepts in multigrid methods. Multigrid methods, which are originally applied as a way to solve elliptic boundary value problems, are regarded as a class of efficient iterative methods to solve various PDEs. Typically their iteration numbers are a constant number and do not depend on the grid sizes. Problems that can be efficiently solved by multigrid methods involve almost all fields of physics and engineering sciences, for instance, elliptic PDEs such as the Poisson's problem, convection diffusion equations, hyperbolic problems as well as clustering and eigenvalue problems [4][9][22]. These problems are difficult to derive the analytic solutions. A practical way to overcome the difficulties is to give numerical solutions by suitable discretization and to solve the corresponding linear or nonlinear large sparse matrix.

In this chapter, we will focus on the discrete Poisson's equation with Dirichlet boundary condition as our model problem:

$$-\Delta_h u_h = f_h, \tag{3.1}$$

in the square Ω_h with $h = \frac{1}{N+1}$, $N \in \mathbb{N}$. We define the finite difference domain Ω_h as:

$$\Omega_h = \Omega \cap G_h, \tag{3.2}$$

where $\Omega = (0,1)^2 \subset \mathbb{R}$ and

$$G_h \equiv \{(x,y) : x = ih, y = jh; i,j \in \mathbb{Z}\}. \tag{3.3}$$

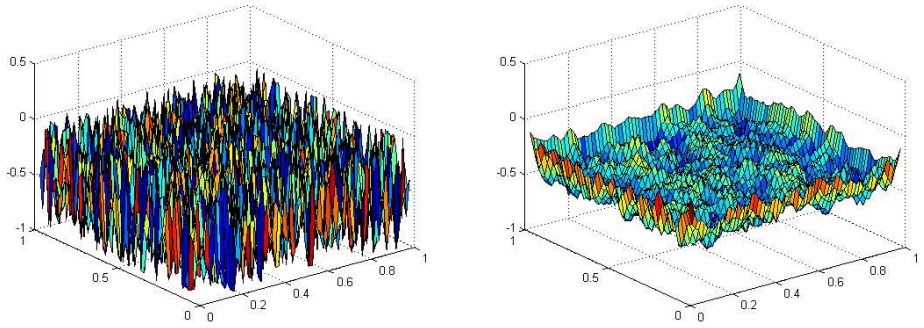
Let L_h be the standard five-point approximation of the partial differential operator, and the stencil is given by:

$$\frac{1}{h^2}\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}_h. \tag{3.4}$$

If we order the unknowns from the lower left corner to the upper right corner, i.e, natural ordering, then the resulting matrix will be a block tridiagonal sparse matrix as in (2.38).

There exist various methods to solve the matrix problem, for example, the Richardson method, the Gauss-Seidel method, the Jacobi method and the successive over-relaxation method. For the model problem (3.1), the convergence rate for the Richardson method is given by $\rho = \frac{\mathcal{K}(A_h)-1}{\mathcal{K}(A_h)+1}$, where $\mathcal{K}(A_h) = \frac{4}{\pi^2 h^2}$ is the condition number of the matrix A_h in (2.38) with the grid size h. The Richardson method converges faster as the condition number is getting closer to 1. When the grid size is refined, the condition number becomes a large number. Thus, the Richardson method results in an inefficient method. The Jacobi and Gauss-Seidel methods have an asymptotic convergence rate of 1 - $O(h^2)$. The optimal successive over-relaxation is one order of magnitude faster than the Jacobi method, which is 1 - $O(h)$. In any case, the convergence of the stationary iterative methods will deteriorate with the grid size refined [10].

The slow convergence of the stationary iterative methods is mainly due to the difficulty in reducing the low frequency components of the errors (we will explain more about high and low frequencies later in this chapter). However, the high frequency errors are damped rapidly in these methods [4]. Figure 3.1 demonstrates the error given by the Gauss-Seidel method applied to the model problem in (3.1). The initial error with a random initial guess shown in Figure 3.1(a) is highly oscillating. Two Gauss-Seidel iterations efficiently damp the high frequency components of the errors, as shown in Figure 3.1(b), in which the errors are relatively smooth.



(a) Initial Errors (b) Error after Pre-smoothing

Figure 3.1: Illustration of high frequency errors reduced by stationary iterative methods. (a) Initial errors with some random initial value, (b) errors after pre-smoother.

In contrast with the standard or stationary iterative methods, the convergence rate of multigrid methods is often independent of the size of the finest grid. The main idea behind multigrid methods is to reduce the low frequency errors by applying a hierarchy of coarse grids. Multigrid methods consist of three important procedures: (1) the pre-smoothing, which is typically an iterative method applied on the current grid aiming to obtain smoothed errors, (2) the coarse grid correction, which computes the error from the coarse grid in order to update the solution on the current grid, (3) the post-smoothing, which is usually the same as the pre-smoothing [4].

In this chapter, we will give an overview and examples of each element of multigrid methods, including pre-smoothing, coarse grid correction procedure, coarsening, coarse grid operator, restriction operator, prolongation operator, post-smoothing, and demonstrate how these parts are organized together.

3.1 Pre-smoothing

The pre-smoothing operator plays a critical role in multigrid methods since it will reduce the high frequency errors significantly, and the smoothed errors are solved on the coarse grids recursively. The stationary iterative methods are commonly used as smoother in the multigrid methods, for instance, the Gauss-Seidel method and the damped-Jacobi method.

3.1.1 Jacobi Iteration

Consider the model problem (3.1). Let $\hat{u}^k_{h,i,j}$ be the approximate solution after k^{th} iteration of the Jacobi method at grid point (ih, jh). The $(k+1)^{th}$ iteration is defined as:

$$\hat{u}^{k+1}_{h,i,j} = \frac{1}{4}[h^2 f_{h,i,j} + \hat{u}^k_{h,i-1,j} + \hat{u}^k_{h,i+1,j} + \hat{u}^k_{h,i,j-1} + \hat{u}^k_{h,i,j+1}], \quad 1 \le i, j \le N. \tag{3.5}$$

Let $\hat{u}^k_h$ be a vector of $\hat{u}^k_{h,i,j}$, $1 \le i, j \le N$, with the lexicographic order. The Jacobi iteration can be expressed as

$$\hat{u}^{k+1}_h = S_h \hat{u}^k_h + \frac{h^2}{4} f_h,$$

with the operator

$$S_h = I_h - \frac{h^2}{4} L_h,$$

where I_h is the identity operator.

If we introduce a damping factor ω, then the damped Jacobi iteration is given by:

$$u_h^{k+1} = \hat{u}_h^k + \omega(\hat{u}_h^{k+1} - \hat{u}_h^k), \tag{3.6}$$

where $\hat{u}_h^{k+1}$ is defined in (3.5). The operator of the damped Jacobi iteration becomes

$$S_h(\omega) = I_h - \frac{\omega h^2}{4} L_h. \tag{3.7}$$

The stencil of $S_h(\omega)$ is given by

$$\frac{\omega}{4} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 4(1/\omega - 1) & 1 \\ 0 & 1 & 0 \end{bmatrix}_h . \tag{3.8}$$

Note that when the damping factor $\omega = 1$, the damped-Jacobi iteration becomes the Jacobi iteration.

In order to measure the convergence properties of the damped-Jacobi method, we first look at the eigenfunctions of $S_h(\omega)$. They are the same as those of L_h, namely,

$$\phi_h^{k,l}(x, y) = \sin k\pi x \sin l\pi y \quad ((x, y) \subset \Omega_h; \quad (k, l = 1, 2, ..., N)), \tag{3.9}$$

and the corresponding eigenvalues

$$\chi_h^{k,l} = \chi_h^{k,l}(\omega) = 1 - \frac{\omega}{2}(2 - \cos k\pi h - \cos l\pi h). \tag{3.10}$$

Let $\bar{u}_h$ be the exact solution to (3.1). Errors after k^{th} and $(k+1)^{th}$ iteration are ϵ_h^k and ϵ_h^{k+1}, respectively

$$\epsilon_h^k = \bar{u}_h - \hat{u}_h^k,$$
$$\epsilon_h^{k+1} = \bar{u}_h - \hat{u}_h^{k+1}.$$

Using the eigenfunctions and eigenvalues of S_h, ϵ_h^k can be written as:

$$\epsilon_h^k = \sum_{k,l=1}^{n-1} \alpha_{k,l} \phi_h^{k,l}. \tag{3.11}$$

With some easy algebra, we can obtain $\epsilon_h^{k+1} = S_h \epsilon_h^k$. Thus, we can write ϵ_h^{k+1} as:

$$\epsilon_h^{k+1} = \sum_{k,l=1}^{n-1} \chi_n^{k,l} \alpha_{k,l} \phi_h^{k,l}. \tag{3.12}$$

We denote $\phi^{k,l}$ to be an eigenfunction of

$$\begin{array}{ll} \text{low frequency,} & \text{if } \max(k,l) \leq N/2, \\ \text{high frequency,} & \text{if } N/2 \leq \max(k,l) \leq N. \end{array}$$

In order to see the smoothing properties of the damped-Jacobi method, we introduce the smoothing factor $\mu(h;w)$ and its supremum μ^* over h of $S_h(\omega)$ in (3.8) as

$$\begin{aligned} &\mu(h;\omega) \equiv \max\{|\chi_h^{k,l}(\omega)| : N/2 \leq \max(k,l) \leq N\}, \\ &\mu^* \equiv \sup_{h \subset \mathcal{H}} \mu(h;\omega), \end{aligned} \tag{3.13}$$

where $\mathcal{H} = \{h = \frac{1}{N+1} |\ N \in \mathbb{N}, N \geq 4\}$. Substituting (3.10) into (3.13), we can get

$$\begin{aligned} &\mu(h;\omega) = \max\{|1 - \frac{\omega}{2}(2 - \cos k\pi h - \cos l\pi h)| : N/2 \leq \max(k,l) \leq N\}, \\ &\mu^*(\omega) = \max\{|1 - \omega/2|, |1 - 2\omega|\}. \end{aligned} \tag{3.14}$$

This shows that there is no smoothing properties for $\omega > 1$ or $\omega \leq 0$. When $\omega = 1, \mu^*(\omega) = 1$, while the choice of $\omega = 4/5$ gives $\mu^*(4/5) = 3/5$. It means all the high frequency error components are reduced by at least a factor of $3/5$ [22]. This explains that the damped Jacobi iteration is efficient to reduce the high frequency error components.

3.1.2 Gauss-Seidel Iteration

The Gauss-Seidel (GS) method is widely used as the smoothing operator in multigrid methods. Consider the same problem in (3.1), and still assume $\hat{u}_h^k$ is the numerical approximate solution after k^{th} iteration. The next iteration becomes:

$$\hat{u}_{h,i,j}^{k+1} = \frac{1}{4}\left[h^2 f_{h,i,j} + \hat{u}_{h,i-1,j}^{k+1} + \hat{u}_{h,i+1,j}^{k} + \hat{u}_{h,i,j-1}^{k+1} + \hat{u}_{h,i,j+1}^{k}\right]. \tag{3.15}$$

By applying the same idea of the damped-Jacobi, we can introduce the damping factor ω here for the GS iteration. Then the step becomes:

$$u_{h,i,j}^{k+1} = \hat{u}_{h,i,j}^{k} + \omega\left(\hat{u}_{h,i,j}^{k+1} - \hat{u}_{h,i,j}^{k}\right), \tag{3.16}$$

where the $\hat{u}_{h,i,j}^{k+1}$ is defined in (3.15). We call this algorithm ω-GS in the following.

For the model problem, the smoothing factor of the GS method is

$$\mu(GS) = 0.5, \quad \text{for } \omega = 1.$$

This implies that the GS method is better at smoothing than the Jacobi method. However, the introduction of a relaxation parameter does not improve the smoothing properties of GS [22].

3.2 Coarse Grid Correction

3.2.1 Multigrid Operators

In this section, we will introduce and give examples to some components of the multigrid.

Coarse Grids

Here we will give a picture of the coarse grids with standard coarsening in Figure 3.2 with finest grid size $h = \frac{1}{32}$ and coarse grid size $h = \frac{1}{16}, h = \frac{1}{8}, h = \frac{1}{4}$, respectively. We assume that $h = \frac{1}{2^p}$, $p \in \mathbb{N}$. Then we can form the grid sequence

$$\Omega_h, \Omega_{2h}, \Omega_{4h}, \Omega_{8h}, ..., \Omega_{h_0},$$

where h_0 is the grid size of the coarsest grid. Specifically, in this paper, the coarse grid domain Ω_H is with coarse grid size $H = 2h$. There are also other coarsening, for instance, semicoarsening [4].

Restriction Operator $\mathcal{R}$ and Prolongation Operator $\mathcal{P}$

In order to inject a grid point on the fine grid to the coarse grid and interpolate the point on the coarse grid back to the fine grid, we need a restriction operator $\mathcal{R}$ and an interpolation operator $\mathcal{P}$, respectively. These two operators are defined as:

$$\mathcal{R} : \Omega_h \to \Omega_H, \quad \mathcal{P} : \Omega_H \to \Omega_h. \tag{3.17}$$

Here we will give two examples of choices of the restriction operator $\mathcal{R}$: injection and averaging.

The injection restriction operator $\mathcal{R}$ is defined as an operator projecting from the fine grid to the coarse grid by taking the corresponding value at the fine grid, i.e.

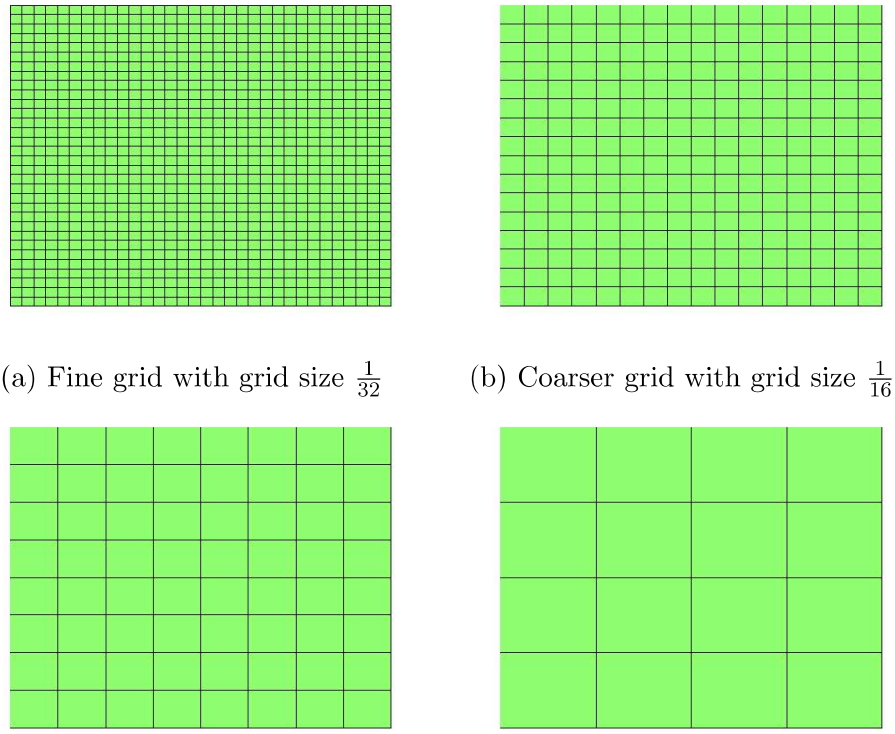
$$u_{H,i^*,j^*} = \mathcal{R}_{injection} u_{h,i,j} = u_{h,i,j}, \tag{3.18}$$

where $(i^*H, j^*H) = (ih, jh) \in \Omega_H \subset \Omega_h$. The stencil of $\mathcal{R}_{injection}$ is:

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}_h^H. \tag{3.19}$$

Averaging operator computes the value at a coarse grid point by taking the weighted average of the corresponding point and its eight neighbours on the fine grid. For the grid point $(i^*H, j^*H) = (ih, jh) \in \Omega_H \subset \Omega_h$, the formula is:

$$\begin{aligned} u_{H,i^*,j^*} =& \mathcal{R}_{averaging} u_{h,i,j} \\ =& \frac{1}{16} \left[u_{h,i-1,j-1} + 2u_{h,i-1,j} + u_{h,i-1,j+1} \right. \\ & + 2u_{h,i,j-1} + 4u_{h,i,j} + 2u_{h,i,j+1} \\ & \left. + u_{h,i+1,j-1} + 2u_{h,i+1,j} + u_{h,i+1,j+1} \right]. \end{aligned} \tag{3.20}$$



(a) Fine grid with grid size $\frac{1}{32}$ (b) Coarser grid with grid size $\frac{1}{16}$

(c) Coarser grid with grid size $\frac{1}{8}$ (d) Coarsest grid with grid size $\frac{1}{4}$

Figure 3.2: Illustration of an hierarchy of grids starting with $h = \frac{1}{32}$.

The stencil of $\mathcal{R}_{averaging}$ is:

$$\frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}_h^H. \tag{3.21}$$

Here we introduce one bilinear interpolation or prolongation operator $\mathcal{P}_{bilinear}$ which maps grid points from the coarse grid Ω_H to the fine grid Ω_h. The formula reads

$$u_{h,i^*,j^*} = \mathcal{P}_{bilinear} u_{H,i,j} = \begin{cases} u_{H,i,j}, & \text{for } \bullet \\ \frac{1}{2}\left[u_{H,i,j+1} + u_{H,i,j-1}\right], & \text{for } \square \\ \frac{1}{2}\left[u_{H,i+1,j} + u_{H,i-1,j}\right], & \text{for } \diamond \\ \frac{1}{4}\left[u_{H,i+1,j} + u_{H,i-1,j}\right. & \\ \quad \left. + u_{H,i,j+1} + u_{H,i,j-1}\right], & \text{for } \circ, \end{cases} \tag{3.22}$$

where $(i^*h, j^*h) \in \Omega_h$. Figure 3.3 presents a fine grid with the symbols for both the fine and the coarse grid points referred by (3.22).

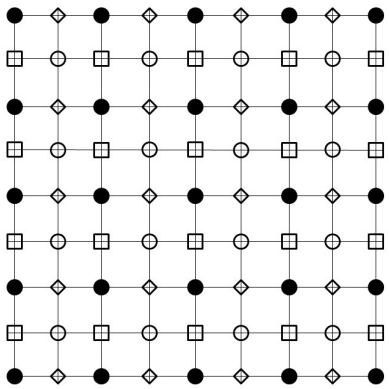


Figure 3.3: A fine grid with symbols presented the bilinear interpolation in (3.22) used for the interpolation from the coarse grid ($\bullet$)

The stencil of $\mathcal{P}_{bilinear}$ is given by:

$$\frac{1}{4}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix}_H^h. \tag{3.23}$$

3.2.2 Coarse Grid Correction

In this section, we will introduce the idea of the coarse grid correction. Given the approximate solution $\tilde{u}_h^k$ after k smoothing steps, we have:

$$\bar{u}_h = \tilde{u}_h^k + \epsilon_h^k, \tag{3.24}$$

where ϵ_h^k is the error. The residual is defined as:

$$r_h^k \equiv f_h - A_h\tilde{u}_h^k = A_h\bar{u}_h - A_h\tilde{u}_h^k = A_h(\bar{u}_h - \tilde{u}_h^k) = A_h\epsilon_h^k. \tag{3.25}$$

It is clear that by solving

$$A_h\epsilon_h^k = r_h^k \tag{3.26}$$

exactly, we will get the error ϵ_h^k, with which we can obtain the exact solution by adding $\tilde{u}_h^k$ to it.

However, solving (3.26) exactly leads to the same computation complexity as solving the linear system itself. One way to resolve the problem is to reduce the computational cost by solving the problem on the coarse grid:

$$A_H\tilde{\epsilon}_H^k = r_H^k, \tag{3.27}$$

where A_H is an approximation of A_h on the coarse grid Ω_H and $r_H^k = \mathcal{R}r_h^k$ is the restricted residual. This makes sense since ϵ_h^k is already smooth after pre-smoothing. With the coarse error $\tilde{\epsilon}_H^k$, we may interpolate it back to the fine grid and update $\tilde{u}_h^k$. The coarse grid correction algorithm is given in Algorithm 1.

Input: $\tilde{u}_h^k$, A_h, f_h
Output: $\tilde{u}_h^{k+1}$
Compute the residual: $r_h^k = f_h - A_h\tilde{u}_h^k$
Restrict the residual to the coarse grid Ω_H: $r_H^k = \mathcal{R}r_h^k$
Obtain $\tilde{\epsilon}_H^k$ by approximately solving the coarse grid problem:$A_H\tilde{\epsilon}_H^k = r_H^k$
Interpolate the error to the fine grid Ω_h: $\tilde{\epsilon}_h^k = \mathcal{P}\tilde{\epsilon}_H^k$
Obtain advanced solution $\tilde{u}_h^{k+1} = \tilde{u}_h^k + \tilde{\epsilon}_h^k$

Algorithm 1: Coarse grid correction

Coarse Grid Operator

When solving (3.27) on the coarse grid Ω_H, we have different choices for the coarse grid operator A_H. One is the direct discretization on Ω_H the same way as on fine grid for A_h. Another choice of the coarse grid operator is the Galerkin operator which is defined as:

$$A_H = \mathcal{R} \cdot A_h \cdot \mathcal{P}, \tag{3.28}$$

where $\mathcal{R}$ and $\mathcal{P}$ are the restriction and interpolation operators mentioned before.

3.3 Post-smoothing

The post-smoothing is usually the same as the pre-smoothing operator but computed after the coarse grid correction in order to obtain a symmetric algorithm.

3.4 Two-grid V-cycle

Given an approximate solution after k^{th} iteration u_h^k, the two-grid V-cycle is to compute u_h^{k+1} by combining the pre-smoothing, coarse grid correction and the post-smoothing. The algorithm is given in Algorithm 2. We call it a V-cycle because the structure of one two-grid cycle looks like the English letter "V", as shown in Figure 3.4(a).

The multigrid method can reduce both high and low frequency errors. Figure 3.5 illustrates the error after the coarse grid correction and the error after the post-smoothing. Together with Figure 3.1, Figure 3.5 shows the performance of one iteration of the two-grid V-cycle.

3.5 Multigrid V-cycle

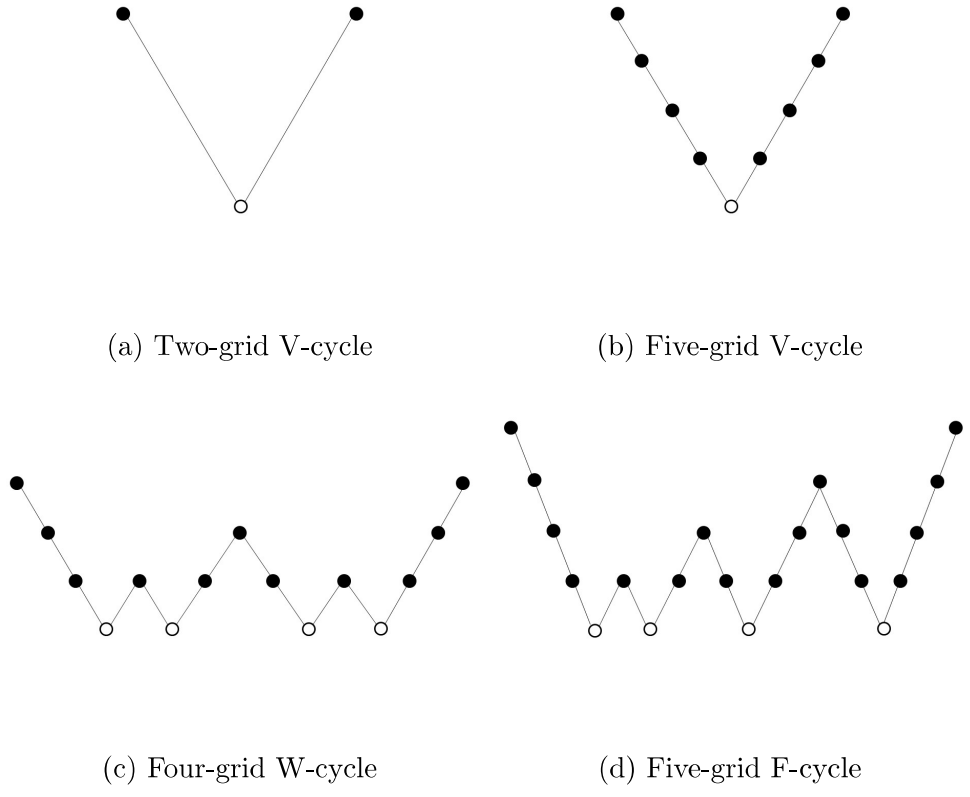
We have described the two-grid V-cycle in the previous section. In practice, the two-grid methods are seldom used, since the problem on the coarse grid is still large. However, they are the basis for the multigrid methods. By applying two-grid V-cycle recursively, we can get a multigrid V-cycle with a hierarchy level of coarse grids. The algorithm is given in Algorithm 3. The level with the biggest grid size is called the coarsest level, in which we solve the problem "exactly". "Exactly" means we still give the numerical solution instead

Input: u_h^k , A_h, f_h
Output: u_h^{k+1} =TwoGridVCycle(u_h^k, A_h, f_h)
if *coarsest grid* **then**
Solve the problem with stationary iterative methods $u_h^{k+1} = \mathcal{S}(u_h^k, A_h, f_h)$;
return u_h^{k+1};
end
else
(1) Pre-smoothing
Compute $\tilde{u}_h^k$ by applying ν_1 times smoothing iterations to u_h^k:
$\tilde{u}_h^k = \mathcal{S}^{\nu_1}(u_h^k, A_h, f_h)$
(2) Coarse Grid Correction
Compute the residual: $\tilde{r}_h^k = f_h - A_h\tilde{u}_h^k$
Restrict the residual to the coarse grid: $\tilde{r}_H^k = \mathcal{R}_r\tilde{r}_h^k$
Compute the error on the coarse grid Ω_H by solving: $A_H\tilde{\epsilon}_H^k = \tilde{r}_H^k$
Interpolate the correction error: $\tilde{\epsilon}_h^k = \mathcal{P}\tilde{\epsilon}_H^k$
Correct the approximation after pre-smoothing: $\hat{u}_h^k = \tilde{u}_h^k + \tilde{\epsilon}_h^k$
(3) Post-smoothing
Compute u_h^{k+1} by applying ν_2 times smoothing iterations to $\hat{u}_h^k$:
$u_h^{k+1} = \mathcal{S}^{\nu_2}(\hat{u}_h^k, A_h, f_h)$
return u_h^{k+1}
end

Algorithm 2: Two grid V-cycle

of really solving the problem, but we solve it with either stationary point iterative methods or other iterative methods to give a relatively accurate numerical solution with the same stopping criterion on the fine grid.

Note that the parameter γ appears twice in (3.29) in Algorithm 3, i.e, γ_h and γ_H. γ_H indicates the cycles to be carried out on the coarser grid. γ_h specifies the number of cycles to be employed on the current grid [22]. If $\gamma = 1$ for all the grid levels, the algorithm is a V-cycle multigrid. The structure of a five-grid V-cycle is shown in Figure 3.4(b).



(a) Two-grid V-cycle

(b) Five-grid V-cycle

(c) Four-grid W-cycle

(d) Five-grid F-cycle

Figure 3.4: Structure of various multigrid cycles for different numbers of grids (•, smoothing, ∘, exact solution, \, fine grid to coarse grid, /, coarse grid to fine grid).

Input: ν_1, ν_2, γ_h, u_h^k , A_h, f_h
Output: $u_h^{k+1} = \text{MultigridVCycle}(\nu_1, \nu_2, \gamma_h, u_h^k, A_h, f_h)$
if *coarsest grid* **then**
Solve the problem with stationary iterative methods $u_h^{k+1} = \mathcal{S}(u_h^k, A_h, f_h)$;
return u_h^{k+1};
end
else
(1) Pre-smoothing
Compute $\tilde{u}_h^k$ by applying ν_1 times smoothing iterations to u_h^k:
$\tilde{u}_h^k = \mathcal{S}^{\nu_1}(u_h^k, A_h, f_h)$
(2) Coarse Grid Correction
Compute the residual: $\tilde{r}_h^k = f_h - A_h\tilde{u}_h^k$
Restrict the residual on coarse grid: $\tilde{r}_H^k = \mathcal{R}_r\tilde{r}_h^k$
Give an initial value for the error on the coarse grid: ϵ_H^k
Compute the error on coarse grid Ω_H γ_h times by recursively solve:

$$\tilde{\epsilon}_H^k = \text{MultigridVCycle}^{\gamma_h}(\nu_1, \nu_2, \gamma_H, \epsilon_H^k, A_H, \tilde{r}_H^k) \tag{3.29}$$

Interpolate the correction error: $\tilde{\epsilon}_h^k = \mathcal{P}\tilde{\epsilon}_H^k$
Correct the approximate solution: $\hat{u}_h^k = \tilde{u}_h^k + \tilde{\epsilon}_h^k$
(3) Post-smoothing
Compute u_h^{k+1} by applying ν_2 times smoothing iterations to $\hat{u}_h^k$:
$u_h^{k+1} = \mathcal{S}^{\nu_2}(\hat{u}_h^k, A_h, f_h)$
return u_h^{k+1}
end

Algorithm 3: Multigrid V-Cycle

3.6 Other Multigrid Cycles

3.6.1 W-cycle

If we let $\gamma = 2$ for all the grid levels, in Algorithm 3, the algorithm becomes a W-cycle multigrid algorithm. This implies that we carry out two cycles of computation on each grid level. The W-cycle is named after the shape of its structure, as shown in Figure 3.4(c). Since the W-cycle visits the coarse grids more than twice, the W-cycle converges generally faster than the V-cycle. However, the computation in one W-cycle is also more expensive than that in one V-cycle.

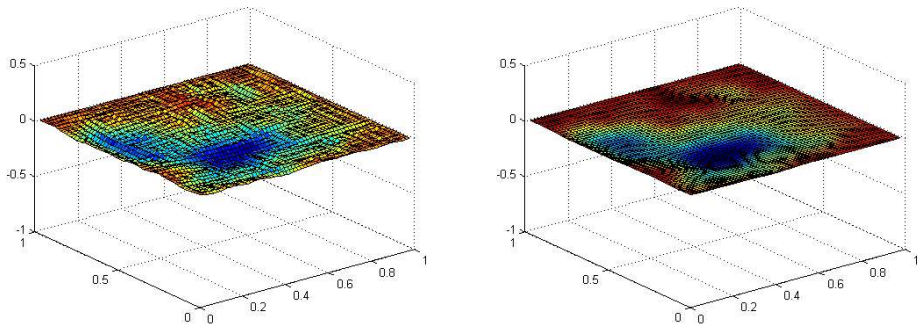
3.6.2 F-cycle

It is convenient and easy to implement if we fix the γ in Algorithm 3.The V-cycle ($\gamma = 1$) and the W-cycle ($\gamma = 2$) are widely used in practice. However, it is not necessary to take γ as a fixed number. Certain combinations of $\gamma = 1$ and $\gamma = 2$ can be chosen in practice. Here, we introduce a so-called F-cycle, the structure of which is illustrated in Figure 3.4(d). The structure of the F-cycle implies that when the algorithm goes down from the fine grid to the coarse grid, $\gamma = 2$. When it goes back from the coarse grid to the fine grid, $\gamma = 1$. Hence, we can regard the F-cycle as a cycle with $1 < \gamma < 2$.

We mentioned that the convergence rate of the multigrid method is independent of the size of its finest grid. However, this fact says nothing about its efficiency unless we take the computational work into account. Table 3.1 shows the computational work for different cycles. In the table, C is a small constant number, and N_L is the total unknowns on the fine grid. From the table, we can conclude that the computational work in one W-cycle is about 1.5 times of that in one V-cycle. The computational work of one F-cycle is between $\frac{4}{3}CN_L$ (V-cycle) and $2CN_L$ (W-cycle) [22].

γ	Computational Work W_L
$\gamma = 1$	$\frac{4}{3}CN_L$
$\gamma = 2$	$2CN_L$
$\gamma = 3$	$4CN_L$
$\gamma = 4$	$O(N_L \log N_L)$

Table 3.1: The computational work of one complete multigrid cycle



(a) Error after CGC (b) Error after post-smoothing

Figure 3.5: Illustration of both high and low frequency errors reduced by the coarse grid correction and the post-smoothing, (a) errors after the coarse grid correction, (b) errors after the post-smoothing

Chapter 4

Multigrid Method for Obstacle Problem

In this chapter, we will introduce a multigrid method, full approximate scheme (FAS), which can be used to solve nonlinear PDEs.

4.1 Model Problem

Consider the lower obstacle problem, where the HJB equation is given in (2.42), as our model problem. Here we let $A_h(u_h) = A_h^M(u_h)$ be the Laplacian operator in (3.4), the matrix derived from the membrane constrained obstacle problem and let $A_h(u_h) = A_h^{MS}(u_h)$ be the minimal surface operator in (2.41). We assume there is no external force and hence f_h is a zero vector for both obstacle problems. In this chapter, we will multiply h^2 on both side of the equation. For example, the Laplacian operator A_h^M becomes

$$A_h^M = \begin{pmatrix} D & B & 0 & \cdots & \cdots \\ B & D & B & \cdots & \cdots \\ 0 & B & D & B & \cdots \\ \vdots & \vdots & \vdots & \ddots & \cdots \\ 0 & \cdots & \cdots & B & D \end{pmatrix}, \tag{4.1}$$

where D and B are defined as in (2.39) and (2.40), respectively.

For simplicity, we can express our model problem in (2.42) as

$$N_h u_h = b_h. \tag{4.2}$$

The matrix N_h can be considered as the merging of the rows of two matrices. One matrix is A_h, and the other is the identity matrix I_h with the same size as A_h. If a grid point in ordering i is active, which means $A_h(u_h)_i - f_{h,i} \geq u_{h,i} - \psi_{h,i}$, we take the i^{th} row from I_h. We let the right hand side $b_{h,i} = \psi_{h,i}$. So the HJB equation becomes $N_h(u_h)_i = u_{h,i} = b_{h,i} = \psi_{h,i}$, that is $u_{h,i} = \psi_{h,i}$. Otherwise, we take the i^{th} row from A_h and let $b_{h,i} = 0$. Thus, the problem becomes $A_h(u_h)_i = 0$. To be more specific, the matrix N_h and right hand side b_h are defined as

$$N_h(i,:) = \begin{cases} A_h(i,:), & \text{if the } i^{th} \text{ grid point is inactive,} \\ I_h(i,:), & \text{otherwise,} \end{cases} \tag{4.3}$$

$$b_{h,i} = \begin{cases} 0, & \text{if the } i^{th} \text{ grid point is inactive,} \\ \psi_{h,i}, & \text{otherwise,} \end{cases} \tag{4.4}$$

where $(i,:)$ means the i^{th} row of a matrix.

We also define an operator $\mathcal{N}_h(u_h) \equiv N_h u_h$. Note that the matrix N_h or the operator $\mathcal{N}_h()$ depends on the approximate solution u_h. In general $\mathcal{N}_h(v_h) \neq \mathcal{N}_h(w_h)$, if $v_h \neq w_h$. Also note that $b_{h,i} = 0$ if the i^{th} grid point on the fine grid is inactive. However, it is in general nonzero on the coarse grids.

4.2 Nonlinear Problem

The model problem in (2.42) can not be solved by the multigrid method in Algorithm 2 and Algorithm 3 mainly because in the multigrid V-cycle, we assume (3.25). But in the nonlinear case,

$$N_h \bar{u}_h - N_h \tilde{u}_h^k \neq N_h(\bar{u}_h - \tilde{u}_h^k), \tag{4.5}$$

in general. We cannot compute the error directly on the coarse grid as in the V-cycle algorithm. So there is no obvious way to compute $\tilde{\epsilon}_h^k$, which makes the V-cycle algorithm not applicable here.

To solve the nonlinear problems, we will introduce another algorithm: FAS. Firstly, we consider

$$\tilde{r}_h^k = b_h - N_h(\tilde{u}_h^k), \tag{4.6}$$

where $\tilde{r}_h^k$ is the residual at k^{th} iteration. Substitute $N_h(\bar{u}_h) = b_h$ into (4.6), we obtain

$$\tilde{r}_h^k = N_h(\bar{u}_h) - N_h(\tilde{u}_h^k). \tag{4.7}$$

Rearrange (4.7), the equation reads,

$$N_h(\bar{u}_h) = \tilde{r}_h^k + N_h(\tilde{u}_h^k). \tag{4.8}$$

If we introduce a restriction operator $\mathcal{R}$ here, we can consider a similar problem on the coarse grid as

$$N_H(\check{u}_H) = \mathcal{R}\tilde{r}_h^k + N_H(\mathcal{R}\tilde{u}_h^k). \tag{4.9}$$

Writing the right hand side $\mathcal{R}\tilde{r}_h^k + N_H(\mathcal{R}\tilde{u}_h^k)$ as b_H, we have

$$N_H(\check{u}_H) = b_H, \tag{4.10}$$

on the coarse grid Ω_H. With the updated $\check{u}_H$, we can compute the error $\tilde{\epsilon}_H$ on the current coarse grid by

$$\tilde{\epsilon}_H = \check{u}_H - \mathcal{R}\tilde{u}_h^k.$$

The remaining steps are similar to the coarse grid correction in the V-Cycle. The algorithm is described in Algorithm 4.

Similar to Algorithm 3, the FAS algorithm also consists of three main components: the pre-smoothing, the coarse grid correction and the post-smoothing. However, the smoothing operators are different from the one in the V-cycle, since the GS method is not applicable to the nonlinear problems. For the coarse grid correction, instead of computing the error on the coarse grid, the FAS solves the solution on the coarse grid. Note that γ in the FAS determines different cycles such as the V-cycle FAS (for $\gamma = 1$) and the W-cycle FAS (for $\gamma = 2$).

4.3 Operators for FAS

Similar to linear multigrid methods, we have different choices for the operators in the FAS. In this section, we give our choice of operators.

4.3.1 Pre-Smoothing and Post-Smoothing

For the smoothing operator of the FAS, we choose the Newton-Gauss-Seidel (NGS) method instead of the Gauss-Seidel (GS) method. Since the HJB equation we are solving is in general not a linear problem, GS is not applicable directly to the nonlinear system.

Before we introduce the NGS method, we first look at the Newton's method. The well known method to solve $\mathcal{F}(x) = 0$ is the Newton's method [15]. Here $\mathcal{F} : \mathbb{R}^n \to \mathbb{R}^n$ is a continuously differentiable operator, i.e., a smooth operator and x is a vector in $\mathbb{R}^n$. The operator $\mathcal{F}$ may depend on the vector x.

Input: $\hat{u}_h^k$, N_h, b_h
Output: $\hat{u}_h^{k+1} = \text{FAS}(\hat{u}_h^k, N_h, b_h)$
(1) Pre-smoothing
Compute $\tilde{u}_h^k$ by applying ν_1 times smoothing iterations to $\hat{u}_h^k$:
$\tilde{u}_h^k = \mathcal{S}^{\nu_1}(\hat{u}_h^k, N_h, b_h)$
(2) Coarse Grid Correction
Compute the residual: $\tilde{r}_h^k = b_h - N_h \tilde{u}_h^k$
Restrict the residual on coarse grid: $\tilde{r}_H^k = \mathcal{R}_1 \tilde{r}_h^k$
Restrict approximate solution on coarse grid: $\tilde{u}_H^k = \mathcal{R}_2 \tilde{u}_h^k$
Compute the right hand side: $b_H = \tilde{r}_H^k + N_H(\tilde{u}_H^k)$
if Ω_H *is the coarsest grid* **then**
    Solve $N_H(\check{u}_H^k) = b_H$ exactly
    return $\hat{u}_h^{k+1}$
end
else
    Solve $N_H(\check{u}_H^k) = b_H$ approximately by applying γ times ($\gamma = 1$ for V-cycle and $\gamma = 2$ for W-cycle) $\check{u}_H^k = \text{FAS}(\tilde{u}_H^k, N_H, b_H)$ recursively
end
Compute correction error: $\tilde{\epsilon}_H^k = \check{u}_H^k - \tilde{u}_H^k$
Interpolate the correction error: $\tilde{\epsilon}_h^k = \mathcal{P}\tilde{\epsilon}_H^k$
Correct the approximation after pre-smoothing: $\hat{u}_h^{k,CGC} = \tilde{u}_h^k + \tilde{\epsilon}_h^k$
(3) Post-smoothing
Compute $\tilde{u}_h^{k+1}$ by applying ν_2 times smoothing iterations to $\hat{u}_{h,CGC}^k$:
$\hat{u}_h^{k+1} = \mathcal{S}^{\nu_2}(\hat{u}_{h,CGC}^k, N_h, b_h)$
return $\hat{u}_h^{k+1}$

Algorithm 4: FAS Algorithm

One step of the Newton's method is given by

$$x^{k+1} = x^k - \mathcal{F}'(x^k)^{-1}\mathcal{F}(x^k), \quad (k = 0, 1, 2, 3....). \tag{4.11}$$

However, if $\mathcal{F}$ is not a smooth operator but a locally Lipschitzian operator, then (4.11) cannot be used anymore.

Let $\partial\mathcal{F}(x^k)$ be the generalized Jacobian of $\mathcal{F}$ at x^k, defined in [6]. One can write (4.11) as

$$x^{k+1} = x^k - \mathcal{V}_k^{-1}\mathcal{F}(x^k), \tag{4.12}$$

where $\mathcal{V}_k \in \partial\mathcal{F}(x^k)$. The Newton's method extends to a non-smooth case by using the generalized Jacobian $\mathcal{V}_k$ instead of the derivative $\mathcal{F}'(x^k)$ [19]. We can rewrite (4.12) as

$$\mathcal{V}_k x^{k+1} = \mathcal{V}_k x^k - \mathcal{F}(x^k). \tag{4.13}$$

It is shown that the local and global convergence results hold for (4.13) in [19] when $\mathcal{F}$ is semismooth.

Solving the linear system (4.13) can be expensive. The NGS method is to solve (4.13) approximately by GS instead. In general, the NGS method we used here can be viewed as a non-smooth version of the standard NGS.

One iteration of NGS for the model problem consists three steps. Assume we get the $\hat{u}_h^k$ after the k^{th} iteration of the NGS. The first step of the $(k+1)^{th}$ iteration is to decide the active and inactive sets based on $\hat{u}_h^k$. We update the matrix N_h and decide the operator $\mathcal{N}_h(\hat{u}_h^k)$ based on the active and inactive sets. The second step is to compute the Jacobian matrix $\mathcal{N}_h'(\hat{u}_h^k)$, which is actually N_h. The last step is to get $\hat{u}_h^{k+1}$ by solving $\mathcal{N}_h'(\hat{u}_h^k)\hat{u}_h^{k+1} = \mathcal{N}_h'(\hat{u}_h^k) - \mathcal{N}_h(\hat{u}_h^k)$ with the GS method.

4.3.2 Restriction Operator $\mathcal{R}$ and Prolongation Operator $\mathcal{P}$

In standard FAS, the restriction operator is the averaging operator. In Algorithm 4, we choose the averaging restriction operator for both $\mathcal{R}_1$ and $\mathcal{R}_2$.

The bilinear interpolation operator is chosen as the prolongation operator in standard FAS. However, this operator is not working well for our model problem. Figure 4.1 shows a 1D obstacle problem example to explain the drawback of the linear interpolation. In the figure, the grid points x_i and x_{i+2} are coarse grid points, and all x_i, x_{i+1} and x_{i+2} are points on the fine grid. The rectangular in red is the obstacle we are considering. Let $\tilde{u}_H$ and $\tilde{u}_h$ to be the approximate solution on the coarse grid and the fine grid, respectively.

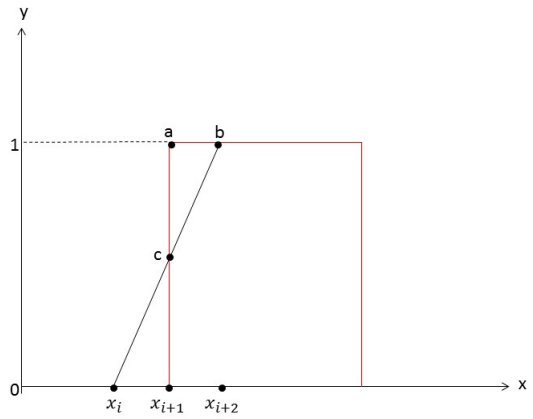


Figure 4.1: 1D obstacle problem example to show the drawback of the linear interpolation

We assume that x_{i+1} and x_{i+2} are active points on the fine grid. So $\tilde{u}_{h,i+1}$ and $\tilde{u}_{h,i+2}$ should be equal to the obstacle value, which is 1 in our example. We also assume $\tilde{u}_{H,i} = 0$. With the linear interpolation, we will get $\tilde{u}_{h,i+1}$ equals to the value at point c instead of 1, which is not accurate.

To solve this problem, we introduce another step after the bilinear interpolation. When we update the approximate solution by adding the error, i.e, the step $\hat{u}_h^{k,CGC} = \tilde{u}_h^k + \tilde{\epsilon}_h^k$ in Algorithm 4, we enforce $\tilde{\epsilon}_{h,i}^k = 0$ if the i^{th} grid point is active. In other words, in the coarse grid correction, the bilinear interpolation is applied only for unknowns on the inactive points. The application of this step results in a better convergence comparing with the pure bilinear interpolation (for details, see M6 in section 5 of [3]).

4.3.3 Nonlinear Operator N_H on Coarse Grid

N_H for Membrane Constrained Obstacle Problem

The standard coarse grid operator of the FAS is the direct discretization. However, this operator results in slow convergence. From Chapter 3, we note that the Galerkin operator works well for the linear problem. Since the Laplacian operator in the membrane constrained problem, i.e. A_h^M in (2.42), is also linear, we can make use of the Galerkin

operator in the nonlinear problem here. This choice results in a faster convergence rate than the direct discretization.

We write the coarse grid HJB equation as

$$\min\left[A_H u_H, u_H - \psi_H\right] = \mathcal{R}\tilde{r}_h^k + N_H \mathcal{R}\tilde{u}_h^k. \tag{4.14}$$

The right hand side of the HJB equation comes from (4.9). For simplicity, the problem can be expressed as

$$N_H u_H = b_H, \tag{4.15}$$

where N_H is the matrix on the coarse grid. N_H is from the merging of A_H and I_H (the identity matrix) in the same way as (4.3).

In order to get the matrix N_H, we need to compute A_H. The Galerkin in (3.28) gives $A_H = \mathcal{R} \cdot A_h \cdot \mathcal{P}$, where A_h is the fine grid matrix. Similarly, we define the matrix A_H in (4.14) as

$$A_H = \mathcal{R} \cdot N_h \cdot \mathcal{P}, \tag{4.16}$$

where N_h is the matrix on the fine grid in (4.3).

The definition of the active and inactive points is the same as that on the fine grid. If an i^{th} point is active, the left hand side of (4.15) becomes $N_H(u_H)_i = I_H(u_H)_i = u_{H,i}$. To make sure the consistency between (4.14) and (4.15), we let $b_{H,i} = \psi_H + \mathcal{R}\hat{r}_h^k + N_H\mathcal{R}\hat{u}_h^k$. The equation becomes $u_{H,i} = \psi_H + \mathcal{R}\hat{r}_h^k + N_H\mathcal{R}\hat{u}_h^k$.

If the point is inactive, the left hand side of (4.15) is $A_H(u_H)_i$, and we let $b_{H,i} = \mathcal{R}\hat{r}_h^k + N_H\mathcal{R}\hat{u}_h^k$. The equation reads $A_H(u_H)_i = \mathcal{R}\hat{r}_h^k + N_H\mathcal{R}\hat{u}_h^k$.

We can get a hierarchy of coarse grid operators by calling (4.16) recursively. This forms a multigrid method. The operator we introduced is quite different from the direct discretization operator, since the matrix A_H in the direct discretization is independent of N_h, the fine grid matrix. But A_H in (4.16) depends on N_h.

N_H for Minimal Surface Obstacle Problem

Since the Galerkin operator results in fast convergence for the membrane constrained obstacle problem, it is natural to apply that to the minimal surface obstacle problem. However, this operator is not working well for the minimal surface obstacle problem.

The Laplacian operator A_h^M in (2.38) for the membrane constrained obstacle problem is independent of the approximate solution u_h. A_H^M should also be independent of u_H on

the coarse grid. We choose the Galerkin operator and define $A_H^M = \mathcal{R} \cdot N_h \cdot \mathcal{P}$. This choice satisfies the independence of the A_H^M on u_H.

On the other hand, the minimal surface operator A_h^{MS} in (2.41) depends on the approximate solution u_h. The operator A_H^{MS} on the coarse grid should also be determined by u_H. However, the Galerkin operator gives $A_H^{MS} = \mathcal{R} \cdot N_h \cdot \mathcal{P}$, which is independent of u_H. So A_H^{MS} given by the Galerkin operator is not accurate for the minimal surface obstacle problem.

Instead, we utilize the direct discretization operator as the coarse grid operator. This operator on the coarse grid depends on u_H and turns out to converge fast.

4.4 FAS Applied on Linear Problem

Here, we explain that the FAS in Algorithm 4 is equivalent to the V-cycle in Algorithm 3 when solving linear problems.

Assume we solve the linear problem (3.1) with the FAS. N_h, N_H is A_h, A_H, respectively, where A_h is a linear operator as defined in (2.38). For the step in the FAS

$$b_H = \tilde{r}_H^k + N_H(\tilde{u}_H^k), \tag{4.17}$$

is now equivalent to

$$b_H = \tilde{r}_H^k + A_H(\tilde{u}_H^k). \tag{4.18}$$

Following the FAS, we obtain $A_H(\check{u}_H^k) = \tilde{r}_H^k + A_H(\tilde{u}_H^k)$. By rearranging the equation, we can get $A_H(\check{u}_H^k - \tilde{u}_H^k) = \tilde{r}_H^k$. Since we update the coarse grid error by $\tilde{\epsilon}_H^k = \check{u}_H^k - \tilde{u}_H^k$, so it is the same as solving $A_H(\tilde{\epsilon}_H^k) = \tilde{r}_H^k$. This is the step in the V-cycle in Algorithm 3. With the coarse grid error, the remaining steps in the FAS and the V-cycle are the same. This proves that applying the FAS to solve linear problem is the same as the V-cycle.

Chapter 5

Numerical Results

In this chapter, we will demonstrate numerical solutions to one-dimensional (1D) and two-dimensional (2D) obstacle problems solved by the V-cycle FAS scheme as proposed in Chapter 4. Two pre-smoothing and post-smoothing iterations are applied. The stopping criterion for all the obstacle problems in this chapter is that the norm of the residual of the problem at the inactive grid point is less than 10^{-6}. More precisely, consider the nonlinear problem $N_h(u_h) = b_h$ and the residual $r_h = N_h(\hat{u}_h) - b_h$. Let

$$r_{h,i}^{in} = \begin{cases} r_{h,i}, & \text{the } i^{th} \text{ grid point is inactive} \\ 0, & \text{otherwise.} \end{cases} \tag{5.1}$$

The stopping criterion is $||r_h^{in}|| < 10^{-6}$. All the 1D examples given in this chapter share the same domain $\Omega = (0, 1) \subset \mathbb{R}$. All the 2D examples given are on $\Omega = (0, 1)^2 \subset \mathbb{R}^2$. Dirichlet boundary conditions are used for all examples.

Here we will give a picture of the obstacles discussed in this chapter. The obstacles considered are the square obstacle (or block obstacle) and the circle obstacle (or dome obstacle). Figure 5.1 shows examples of 1D and 2D obstacles.

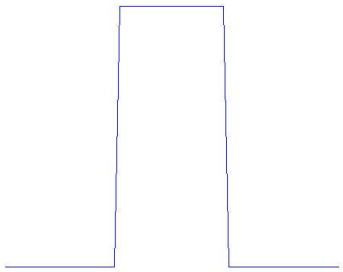
5.1 1D Block Obstacle Problem with Laplacian Operator

We consider a 1D lower obstacle problem subject to a lower block obstacle

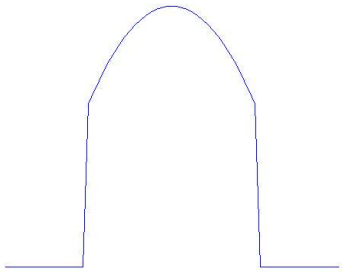
$$u(x) \geq \psi(x) = \begin{cases} 1, & \frac{11}{32} \leq x \leq \frac{21}{32}, \\ 0, & \text{otherwise.} \end{cases} \tag{5.2}$$

As shown in Chapter 2, the HJB equation corresponding to the 1D discrete obstacle problem can be written as

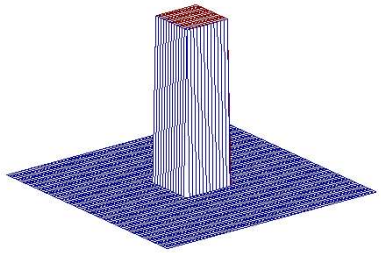
$$\min[A_h u_h - f_h, u_h - \psi_h] = 0, \tag{5.3}$$



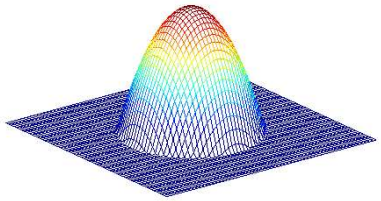
(a) 1D block obstacle



(b) 1D dome obstacle



(c) 2D block obstacle



(d) 2D dome obstacle

Figure 5.1: Pictures of different obstacles.

Grid Size(h)	Levels	FAS Iterations
$\frac{1}{8}$	2	3
$\frac{1}{16}$	3	4
$\frac{1}{32}$	4	5
$\frac{1}{64}$	5	4
$\frac{1}{128}$	6	5

Table 5.1: Iterations of the FAS for 1D block problem

where A_h is a tridiagonal symmetric positive definite matrix as

$$A_h = \begin{pmatrix} 2 & -1 & 0 & \cdots & \cdots \\ -1 & 2 & -1 & \cdots & \cdots \\ 0 & -1 & 2 & -1 & \cdots \\ \vdots & \vdots & \vdots & \ddots & \cdots \\ 0 & \cdots & \cdots & -1 & 2 \end{pmatrix}, \tag{5.4}$$

u_h is the discretized approximate solution and f_h is a zero vector.

We will apply the V-cycle FAS (which is given in Algorithm 4) to solve the HJB equation. Table 5.1 shows the FAS iterations for different grid sizes and levels. The levels here mean the number of levels of fine plus coarse grids in the whole computation. For example, if we say the grid size $\frac{1}{64}$, level 5, it means we have a total of 5 levels, the grid size of each is: $\frac{1}{64}$, $\frac{1}{32}$, $\frac{1}{16}$, $\frac{1}{8}$, $\frac{1}{4}$. Figure 5.2 shows the shape of the obstacle and numerical solution of this 1D obstacle problem.

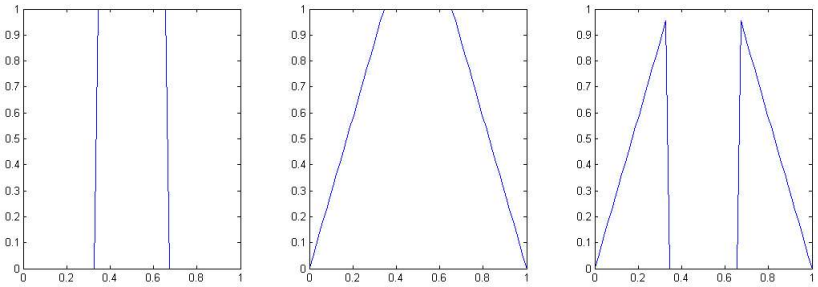


Figure 5.2: 1D block obstacle result. (left) Obstacle, (middle) numerical solution, (right) difference between the obstacle and the numerical solution, with grid size $\frac{1}{64}$ and 5 levels

Grid Size(h)	Levels	FAS Iterations
$\frac{1}{8}$	2	1
$\frac{1}{16}$	3	5
$\frac{1}{32}$	4	7
$\frac{1}{64}$	5	5
$\frac{1}{128}$	6	6

Table 5.2: Iterations of the FAS for 1D dome problem

5.2 1D Dome Obstacle Problem with Laplacian Operator

The HJB equation of the 1D dome obstacle problem is similar to the 1D block obstacle except for the obstacle function. However, for the block obstacle the membrane will contact with the obstacle exactly on the surface of the obstacle. For the dome obstacle, it is not the case. Figure 5.3 (right) shows the difference between the numerical solution and the obstacle, from which we can see that the membrane is not stick to the obstacle exactly, but leaving several grid points on the boarder of the obstacle inactive. Since the active set and inactive set depend on not only the shape of the dome obstacle, but also the approximate solution, the obstacle problem with the dome obstacle is more difficult than the one with the block obstacle.

The problem we consider here is with the constraints

$$u(x) \geq \psi(x) = \begin{cases} 1 - 6(x - 0.5)^2, & \frac{31}{128} \leq x \leq \frac{97}{128}, \\ 0, & \text{otherwise.} \end{cases} \tag{5.5}$$

The dome obstacle problem share the same HJB equation with (5.3), except the obstacle function ψ is given by (5.5). Using the V-cycle FAS, the iterations are presented in Table 5.2. The obstacle, numerical solution and the difference between the numerical solution and the obstacle are illustrated in Figure 5.3. The difference between the numerical solution and the obstacle demonstrates the contact parts between the obstacle and the approximate solution.

Table 5.1 and Table 5.2 show that the iterations of the FAS to solve the 1D block or dome obstacle problem are independent of the finest grid size, and close to a constant number. For instance, for the 1D block obstacle problem, the iteration numbers of different grid sizes are about 4 to 5.

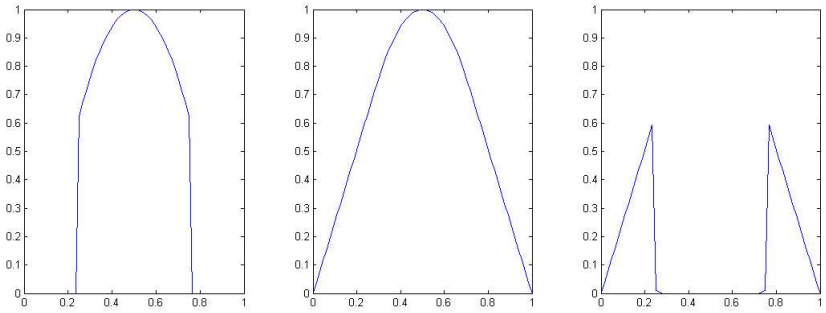


Figure 5.3: 1D Dome obstacle result, with grid size $\frac{1}{64}$ and 5 levels. (left) Obstacle, (middle) numerical solution, (right) difference between the numerical solution and the obstacle

Grid Size(h)	Levels	FAS Iterations
$\frac{1}{8}$	2	4
$\frac{1}{16}$	3	6
$\frac{1}{32}$	4	6
$\frac{1}{64}$	5	7
$\frac{1}{128}$	6	9

Table 5.3: Iterations of the FAS for 2D square problem

5.3 2D Square Obstacle Problem with Laplacian Operator

The example of 2D obstacle problem we considered here is the model problem in (3.1). The obstacle discussed here is

$$u(x,y) \geq \psi(x,y) = \begin{cases} 1, & \frac{13}{32} \leq x \leq \frac{19}{32}, \frac{13}{32} \leq y \leq \frac{19}{32}, \\ 0, & \text{otherwise.} \end{cases} \tag{5.6}$$

Using the same discretization, the matrix form as (4.1) and the HJB equation in (2.42), we apply the FAS to solve this problem iteratively, the iterations are given in Table 5.3. Figure 5.4 shows the 2D square obstacle, the numerical solution and the contact information between the numerical solution and the obstacle.

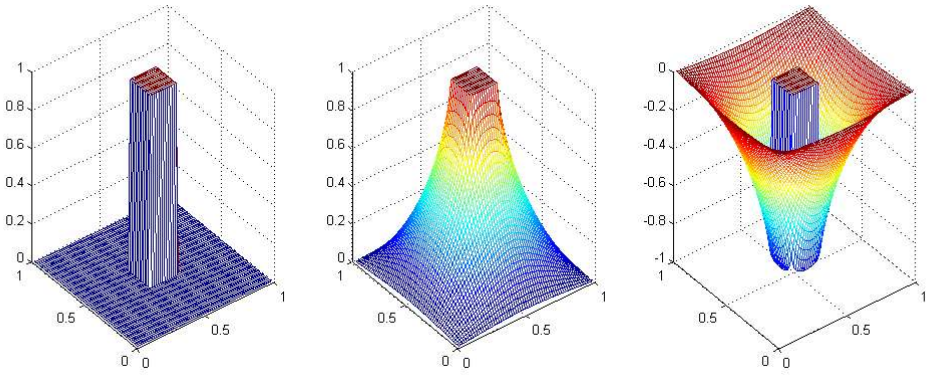


Figure 5.4: 2D Square obstacle result, with grid size $\frac{1}{64}$ and 5 levels. (left) Obstacle, (middle) numerical solution, (right) difference between the numerical solution and the obstacle

5.4 2D Dome Obstacle Problem with Laplacian Operator

In this section, we will give numerical examples of the 2D dome obstacle problem. Similar to the 2D square obstacle problem, we use the same HJB equation with the square obstacle problem but the constraints for the dome obstacle are

$$u(x,y) \geq \psi(x,y) = \max\{0, 0.6 - 8|(x-0.5)^2 + (y-0.5)^2|\}. \tag{5.7}$$

The iterations of the FAS are similar to those of the square obstacle problem as shown in Table 5.4. Figure 5.5 shows the obstacle, the solution and the difference for the 2D dome obstacle problem.

We will compare our method with the methods proposed in [13] and [17]. They both solve the elasto-plasto torsion problem which is similar to our numerical example here. The problem considered in both paper is on the domain $\Omega = (0,1)^2$ with the same right hand side value. The only difference of the elasto-plasto torsion problem compared with our example is the shape of the obstacle. The obstacle considered in both paper leaves the inactive set a small region with a cross shape aligned with two diagonals.

In [13], finite element methods are used as the discretization method and monotone multigrid methods are applied. To compare our method with that in [13], we define the iterative error as $\epsilon_h^k = \bar{u}_h - \hat{u}_h^k$, where $\hat{u}_h^k$ is the approximate solution after the k^{th} iteration.

Grid Size(h)	Levels	FAS Iterations
$\frac{1}{8}$	2	5
$\frac{1}{16}$	3	5
$\frac{1}{32}$	4	6
$\frac{1}{64}$	5	7
$\frac{1}{128}$	6	9

Table 5.4: Iterations of the FAS for 2D dome problem

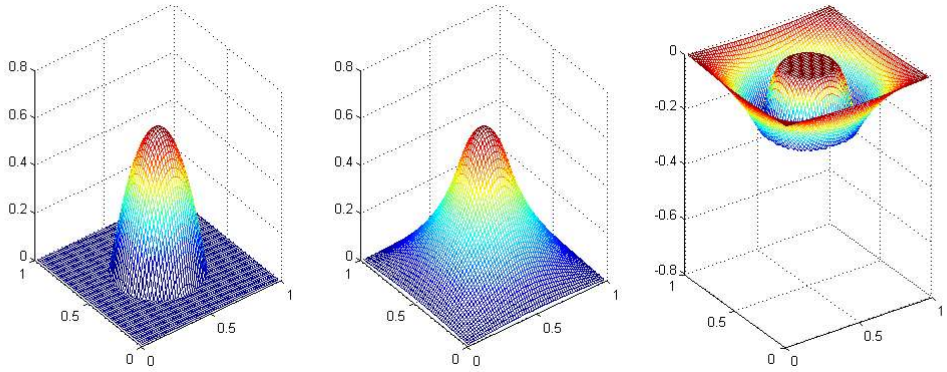


Figure 5.5: 2D Dome obstacle result, with grid size $\frac{1}{64}$ and 5 levels. (left) Obstacle, (middle) numerical solution, (right) difference between numerical solution and obstacle

$\bar{u}_h$ is the "exact" numerical solution computed first by some methods with stopping criterion $||r_h^{in}|| < 10^{-9}$. The comparison of the iteration numbers to achieve different iterative errors of the FAS and the TRCKH in [13] is shown in Table 5.5. From the table, the iterations for the FAS are less than that for the TRCKH. The FAS needs roughly half iterations of the TRCKH to achieve the same iterative error. Since the V-cycle multigrid method are used in both the TRCKH and our FAS, the computations in each iteration of both methods are in the same order. We can conclude that the V-cycle FAS is better than the TRCKH at reducing the iterative errors.

In [17], a multigrid F-cycle is presented. Three iterations of recombinations are applied after each F-cycle. In each recombination, a linear combination of six latest approximate solutions $\hat{u}_h^{k-i}$, $i = 0, 1, 2, 3, 4, 5$, is computed with some special parameters. The special parameters are determined in such a way that the residual of the current problem is minimized. The recombination procedure is not expensive if the parameters are chosen

Iterative Error	10^{-2}	10^{-4}	10^{-6}	10^{-7}	10^{-8}
Number of Iterations for TRCKH in [13]	10	19	23	25	28
Number of Iterations for FAS	8	10	11	12	13

Table 5.5: Comparison of iterative errors

	F-cycle in [17]	FAS
$h = \frac{1}{128}$	0.20	0.098
$h = \frac{1}{256}$	0.37	0.186

Table 5.6: Comparison of the convergence rate between the F-cycle and the FAS

properly. However, the computation in each F-cycle is more expensive than that in one V-cycle, as explained in Chapter 3. Table 5.6 shows the convergence rate for the F-cycle and our V-cycle FAS, where h is the grid size. We can see from the table that for the two grid sizes, the FAS converge faster than the F-cycle in [17].

5.5 2D Dome Obstacle Problem with Minimal Surface Operator

Now, we consider an obstacle problem with the minimal surface operator. The obstacle we considered here is the same as for the 2D obstacle problem with the Laplacian operator, that is (5.7). The convergence results for solving the corresponding HJB equations with the V-cycle FAS algorithm are given in Table 5.7. Figure 5.6 shows the obstacle, the numerical solution and the difference between them. From the figure, we can see that the solution of the minimal surface obstacle problem contacts more tightly with the obstacle.

Table 5.8 shows the residuals for grid size $h = \frac{1}{16}$ and $h = \frac{1}{32}$ in each iteration computed by the FAS. For comparison, we have listed the residuals reported in [11] and [12]. It is noticed that the residuals in both [11] and [12] are reduced faster than the FAS method. However, the W-cycle multigrid is applied in both papers. Consider the computation in each W-cycle is about 1.5 times of that in one V-cycle (cf. Table 3.1), the total computational work of 9 iterations of our FAS is about the same as 6 iterations of the W-cycle. From the table, we see that at the 9^{th} iteration of our method, the residual is reduced to 8.0E - 9 (for grid size $\frac{1}{32}$). However, for the methods in [11] and [12], the residual after

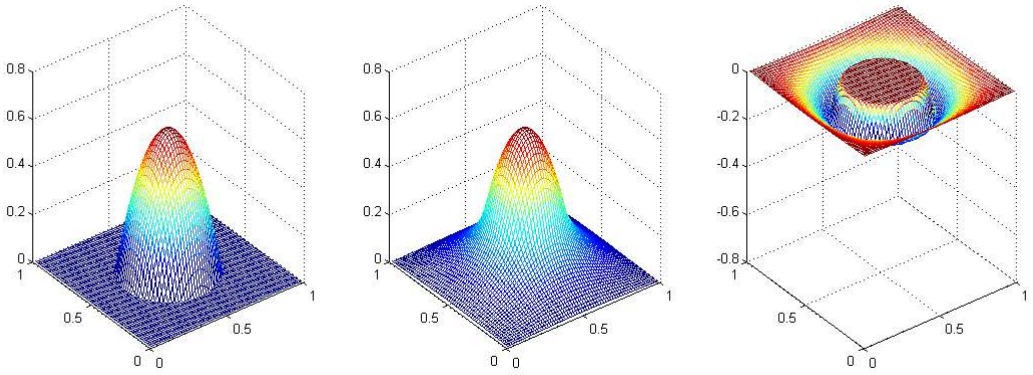


Figure 5.6: 2D circle obstacle with minimal surface operator, with grid size $\frac{1}{64}$ and 5 levels. (left) Obstacle, (middle) numerical solution, (right) difference between the numerical solution and the obstacle

Grid Size(h)	Levels	FAS Iterations
$\frac{1}{8}$	2	4
$\frac{1}{16}$	3	6
$\frac{1}{32}$	4	11
$\frac{1}{64}$	5	10
$\frac{1}{128}$	6	15

Table 5.7: Iterations of the FAS for 2D dome problem with the minimal surface operator

the 6^{th} iteration is reduced to 5.9E - 7 and 2.2E - 7, respectively. From the table, we can conclude that the V-cycle FAS converges faster than the W-cycle in [11] and [12].

It is claimed in [21] that any choice of decompositions of obstacle function ψ ends up with the same convergence rate. However, the number of the domain decompositions may affect the rate of convergence. Table 5.9 presents the convergence rates for FAS we used to solve the minimal surface problem and the space decomposition and subspace correction algorithms used in [21]. In the table, J is the number of decompositions of some domain decomposition methods. We note that the numerical experiment given in [21] is not exactly the same. A slightly different obstacle on the domain $\Omega = (-2, 2)^2$ is applied in [21]. However, it is quite similar to our example because of the same shape of the obstacle and the size of the fine grid, that is $h = L/128$, L = 4 here. From the table, we can conclude that the FAS converges faster than the domain decomposition method in

Iteration	FAS	FAS	vimII in [12]	vimII in [12]	mgm in [11]	mgm in [11]
	$h = 1/16$	$h = 1/32$	$h = 1/16$	$h = 1/32$	$h = 1/16$	$h = 1/32$
1	1.6E - 2	3.7E - 2				
2	1.7E - 3	4.1E - 3			1.9E - 2	
3	2.4E - 4	8.2E - 4	2.4E - 4		1.3E -3	1.1E -3
4	3.7E - 5	3.1E - 4	2.3E -5	1.1E - 3	9.4E - 5	3.6E - 4
5	5.9E - 6	1.2E - 4	2.2E - 6	4.2E - 4	7.2E - 6	1.4E - 4
6	9.7E - 7	4.9E - 5	2.2E - 7	1.5E - 4	5.9E - 7	5.6E - 5
7	1.7E - 7	2.0E - 5	2.0E - 8	5.3E - 5	5.3E - 8	2.3E - 5
8	3.6E - 8	8.1E - 6	5.0E - 9	1.9E - 5		
9	8.0E - 9	3.3E - 6	1.3E - 9	6.7E - 6		
10	1.9E - 9	1.4E - 6	3.2E - 10	2.4E - 6		

Table 5.8: Residuals after each iteration of different methods

	J = 5	J = 6	J = 7	J = 8	FAS
Convergence Rate	0.78	0.8	0.81	0.85	0.40

Table 5.9: Convergence rates for the domain decomposition method in [21] (Column 2 to Column 5) and the FAS (last Column)

[21].

Chapter 6

Conclusion

In this paper, we proposed a multigrid method based on the full approximate scheme (FAS). We used this method to solve two types of obstacle problems in the HJB equation formulations. The Galerkin operator was often used as a coarse grid operator for linear problems. Here we proposed a special coarse grid operator based on the Galerkin approach for solving a nonlinear problem, i.e. the membrane constrained obstacle problem. This choice improved the convergence of the FAS. For the minimal surface obstacle problem, we applied the direct discretization operator as the coarse grid operator. In addition, we proposed a special prolongation operator based on the bilinear operator. This special operator solved the inaccurate value problem at the boundary between the active and inactive set. Numerical results showed that our FAS converges in small numbers of iterations for those obstacle problem examples. The FAS is better at reducing the iterative errors, residuals and converges faster than some other multigrid methods mentioned in the paper. Possible future work includes the proof of the convergence of the FAS.

References

[1] Asma Al-Tamimi, Frank L Lewis, and Murad Abu-Khalaf. Discrete-time nonlinear hjb solution using approximate dynamic programming: convergence proof. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 38(4):943–949, 2008.

[2] Olivier Bokanowski, Stefania Maroso, and Hasnaa Zidani. Some convergence results for howard's algorithm. *SIAM Journal on Numerical Analysis*, 47(4):3001–3026, 2009.

[3] Achi Brandt and Colin W Cryer. Multigrid algorithms for the solution of linear complementarity problems arising from free boundary problems. *SIAM journal on scientific and statistical computing*, 4(4):655–684, 1983.

[4] William L Briggs, Steve F McCormick, et al. *A multigrid tutorial*, volume 72. SIAM, 2000.

[5] Luis A Caffarelli. The obstacle problem revisited. *Journal of Fourier Analysis and Applications*, 4(4):383–402, 1998.

[6] F Clarke. Optimization and nonsmooth analysis wiley. *New York*, 1983.

[7] Isabel Narra Figueiredo, José Francisco Rodrigues, and Lisa Santos. *Free Boundary Problems: Theory and Applications*, volume 154. Springer, 2007.

[8] R Glowinski, Jacques Louis Lions, and Raymond Trémoliéres. *Numerical analysis of variational inequalities*, 1981.

[9] Roland Glowinski. *Lectures on Numerical Methods for Non-Linear Variational Problems*. Springer, 2008.

[10] David Gottlieb and Steven A Orszag. *Numerical analysis of spectral methods*, volume 2. SIAM, 1977.

[11] Wolfgang Hackbusch and Hans Detlef Mittelmann. On multi-grid methods for variational inequalities. *Numerische Mathematik*, 42(1):65–76, 1983.

[12] Ronald HW Hoppe. Multigrid algorithms for variational inequalities. *SIAM journal on numerical analysis*, 24(5):1046–1065, 1987.

[13] Ralf Kornhuber. Monotone multigrid methods for elliptic variational inequalities I. *Numerische Mathematik*, 69:167–167, 1994.

[14] Hans Petter Langtangen. *Computational partial differential equations: numerical methods and diffpack programming*. Springer Berlin, 1999.

[15] Jorge J Moré. Global convergence of newton-gauss-seidel methods. *SIAM Journal on Numerical Analysis*, 8(2):325–336, 1971.

[16] Dianne P O'Leary. Conjugate gradient algorithms in the solution of optimization problems for nonlinear elliptic partial differential equations. *Computing*, 22(1):59–77, 1979.

[17] Cornelis W Oosterlee. On multigrid for linear complementarity problems with application to American-style options. *Electronic Transactions on Numerical Analysis*, 15(165-185):2–7, 2003.

[18] JONG-SHI PANG. Newton's method for b-differentiable equations. *Mathematics of operations research*, 15(2):311–341, 1990.

[19] Liqun Qi and Jie Sun. A nonsmooth version of newton's method. *Mathematical programming*, 58(1-3):353–367, 1993.

[20] J-F Rodrigues. *Obstacle problems in mathematical physics*, volume 134. North Holland, 1987.

[21] Xue-Cheng Tai. Rate of convergence for some constraint decomposition methods for nonlinear variational inequalities. *Numerische Mathematik*, 93(4):755–786, 2003.

[22] Ulrich Trottenberg and Anton Schuller. *Multigrid*. Academic Press, Inc., 2000.