Multigrid for Elliptic Monge Ampère Equation

by

Ashmeen Kaur Soin

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, 2011

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

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

Abstract

We propose a multigrid method to solve elliptic Monge-Ampère equations (MAE). This method is based on full approximation scheme (FAS). The motivation for developing a solver for MAE comes from its application in image registration. Though there are many numerical methods proposed to solve these equations, the convergence is typically slow for large problems. A second order discretization is used to develop the FAS. Performance of the proposed method is tested on problems varying from smooth to mildly singular. Comparison is drawn with the existing relaxation scheme in terms of the computation time.

Acknowledgements

I am deeply thankful to my supervisor, Justin W. L. Wan, for guiding me through this report. He has always been very patient while offering insights to a problem and tried to explain in the most intuitive way possible. I would also like to thank my husband for his love and support and my classmates for their help.

Dedication

This is dedicated to my husband, Sukhmeet Singh Soin.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Monge-Ampère equations were first studied by Gaspard Monge in 1784 [3] and later by Andre-Marie Ampère. Monge-Ampère equations (MAE) have applications in the areas of differential geometry, the calculus of variations, and several optimization problems, such as the Monge-Kantorovich mass transfer problem [3, 16].

1.1 Motivation

The motivation for this paper comes from the application of MAE in image registration. Image registration is a process of aligning data in multiple images obtained by possibly different imaging sources (e.g., X-ray, CT scan, MRI, etc.). This is an important technique in the field of medical imaging. Among many applications, it is used to provide critical pre-operative and intra-operative information.

One approach to find an optimal mapping for the image registration problem, is to apply the Monge-Kantorovich problem of mass transport [14, 13]. To understand the application of MAE, let us consider two images given by $I_1(x)$ and $I_2(x)$, $x \in \mathbb{R}^d$. Find a one to one mapping M from $\mathbb{R}^d$ to $\mathbb{R}^d$ which satisfies

$$\int_{M^{-1}(A)} I_1(x)dx = \int_A I_2(x)dx, \qquad A \subset \mathbb{R}^d \tag{1.1}$$

to preserve the mass while aligning the two images [15]. The L^p Kantorovich-Wasserstein

metric to define distance between $I_1(x)$ and $I_2(x)$ is given by

$$d(I_1, I_2) = \inf \int ||x - M(x)||^p \, I_1(x) \, \mathrm{d}x, \tag{1.2}$$

where $M(x)$ satisfies (1.1). The optimal mapping $\bar{M}$, which minimizes this integral gives the optimal transfer of mass from I_1 to I_2. Thus, it represents the solution of the image registration problem.

Moreover, the mapping $\bar{M}$ can be given by, $\bar{M} = \nabla\psi$, where ψ satisfies the following Monge-Ampère equation

$$det(D^2\psi(x)) = \frac{I_1(x)}{I_2(\nabla\psi)}.$$

Here $det(D^2\psi(x))$ denotes the determinant of the Hessian of ψ. In $\mathbb{R}^2$, this can be equivalently written as:

$$det(D^2\psi(x)) = \frac{\partial^2\psi}{\partial x^2}\frac{\partial^2\psi}{\partial y^2} - \frac{\partial^2\psi}{\partial x \partial y}.$$

The importance of MAE in this problem formulation motivates us to develop a fast solver for these nonlinear equations.

1.2 Monge-Ampère Equation

The Monge-Ampère equation is a nonlinear second order partial differential equation (PDE). The general form of MAE is

$$u_{xx}u_{yy} - u_{xy}^2 = au_{xx} + 2bu_{xy} + cu_{yy} + f, \qquad on\ \Omega \tag{1.3}$$

where the coefficients a,b,c depend on variables x, y, the unknown function u, and its derivatives u_x, u_y.

In general, to determine the type of a nonlinear PDE, $F(D^2u, Du, u, x) = 0$, one linearizes F in the variables of its arguments. The type of the equation is then given by the type of PDE of the linearized operators [11]. In case of MAE this exercise is reduced to determining the sign of the expression $\Theta \equiv f + ac + b^2$. $\Theta > 0$ corresponds to an elliptic type; $\Theta = 0$ to a parabolic type; and $\Theta < 0$ to a hyperbolic type [3].

It is known that if the domain, Ω is not strictly convex, then no general classical solution exists for Dirichlet problems of fully nonlinear second order PDEs [11]. Futhermore, there

may exist multiple other solutions, (e.g. concave) to the MAE, however, for a Dirichlet problem, a strictly positive f ensures existence of a unique convex solution [1, 4].

From [16], we know that the solution u must be convex for the Monge-Ampère equation to be elliptic. The elliptic MAEs do not have a unique solution without the convexity constraint.

We know that solving a nonlinear PDE analytically for solutions can be very complicated. So it is natural to consider numerical methods to approximate the solutions of MAE. There are a number of published papers devoted to solving MAE numerically. Dean and Glowinski [8, 7, 6] have proposed schemes based on Lagrangian and least square methods. However, their method does not converge for problems with moderate singularities. Oberman in [17] introduced a convergent finite difference scheme to solve MAE. Furthermore, in [16] Benamou, Froese, and Oberman propose two second order methods to solve elliptic MAE with possible singularities. We will discuss these two methods in the report subsequently.

1.3 Discussion of content

In this paper, we study the elliptic Monge-Ampère equations on a square domain, $\Omega \subset \mathbb{R}^2$, with a positive source term $f : \Omega \to \mathbb{R}$ and Dirichlet boundary conditions. This paper adopts a finite difference discretization presented by Jean-David Benamou, Brittany D. Froese, and Adam M. Oberman in [16]. In [16] this discretization was used to develop an iterative method which performs well in case of generic (possibly singular) problems, an area which most of the previous methods did not succeed in. However, the proposed iterative method exhibits slow convergence as the size of the problem increases. We know that for the multigrid methods, the convergence usually is independent of the mesh size. Therefore exploring multigrid to develop a fast and efficient method to solve the MAE appears to be the obvious next step.

The contents of this paper are organized as follows. Chapter 2 gives a detailed description of two discretizations for the elliptic MAE, followed by discuss on viscosity solution for MAEs and monotone finite difference schemes. Chapter 3 provides an overview of multigrid method, followed by our development of Multigrid for MAE in Chapter 4. In Chapter 5 we implement our method on several example Monge-Ampère equations and discuss the results. In Chapter 6 we discuss the conclusions drawn.

Chapter 2

Discretization of Monge-Ampère Equation

In this chapter we discuss two discretizations of the elliptic Monge-Ampère equation (1.3) on a uniform grid Ω_h with grid size h. In the first two sections, we discuss the mathematical formulation of a second and fourth order discretizations, respectively. In the third section we discuss viscosity solutions for MAE and a monotone finite difference scheme proposed by Oberman in [17].

2.1 First Discretization

First discretization was proposed by Benamou, Froese, and Oberman [16]. It involves standard central differencing for second derivatives u_{xx}, u_{yy}, and u_{xy}. The elliptic Monge-Ampère equation considered in this paper is given by

$$det(D^2u) = f \quad in \quad \Omega \subset \mathbb{R}^2, \tag{2.1}$$

$$u = g \quad on\ \partial\Omega \tag{2.2}$$

or equivalently,

$$u_{xx}u_{yy} - u_{xy}^2 = f(x,y)$$
$$u = g \quad on\ \partial\Omega \tag{2.3}$$

where Ω is a square $[0,1] \times [0,1]$ in xy-plane. We can see that (2.1) can be obtained from the general MAE (1.3) by setting $a = b = c = 0$.

In the domain Ω, consider a uniform sequence of N grid points along x and y axes denoted as x_i, y_j, where $i, j = 1, 2, ..., N$, respectively. We refer to this discretized domain as Ω_h, where $h = \frac{1}{N-1}$ is the mesh size. Furthermore, consider an arbitrary grid point given by (x_i, y_j). The objective is to approximate the value of the solution $u(x_i, y_j)$, denoted by $u_{i,j}$, for $1 \leq i, j \leq N$.

Now, consider the solution values at grid points in a 9×9 neighborhood of the discrete domain Ω_h

$$\begin{bmatrix} u_{i-1,j+1} & u_{i,j+1} & u_{i+1,j+1} \\ u_{i-1,j} & u_{i,j} & u_{i+1,j} \\ u_{i-1,j-1} & u_{i,j-1} & u_{i+1,j-1} \end{bmatrix}.$$

Discretizing the second derivatives from (2.1), u_{xx}, u_{yy}, u_{xy}, using central differencing on the uniform grid, we obtain

$$(D^2_{xx}u_{i,j})(D^2_{yy}u_{i,j}) - (D^2_{xy}u_{i,j})^2 = f_{i,j}, \tag{2.4}$$

where the finite difference operators are given by

$$\begin{aligned} D^2_{xx}u &= \frac{1}{h^2}[u_{i-1,j} - 2u_{i,j} + u_{i+1,j}], \\ D^2_{yy}u &= \frac{1}{h^2}[u_{i,j+1} - 2u_{i,j} + u_{i,j-1}], \\ D^2_{xy}u &= \frac{1}{4h^2}[u_{i+1,j+1} + u_{i-1,j-1} - u_{i+1,j-1} - u_{i-1,j+1}]. \end{aligned} \tag{2.5}$$

From (2.4) and (2.5), we have

$$\begin{aligned} &\frac{1}{h^4}(u_{i+1,j} + u_{i-1,j} - 2u_{i,j})(u_{i,j+1} + u_{i,j-1} - 2u_{i,j}) \\ &\quad - \frac{1}{4^2h^4}(u_{i+1,j+1} + u_{i-1,j-1} - u_{i-1,j+1} - u_{i+1,j-1})^2 = f_{i,j}. \end{aligned} \tag{2.6}$$

Equation (2.6) is a quadratic in $u_{i,j}$. Since we are using centered differencing for the derivatives, the discritization in (2.6) is second order accurate. Relaxation scheme based on (2.6) performs well for smooth to singular problems.

Furthermore, in the same paper [16], the authors discuss another discretization based on the Laplacian of u, which involves solving a Poisson equation. However, this discretization converges very slowly for a non-smooth solution. Thus we will not discuss this approach here.

2.2 Second Discretization

This second discretization was obtained by Chen in 2009 [3] from the truncation analysis of the errror associated with the discrete MAE.

In [3], they consider a Monge-Ampère equation given by

$$(1 - u_{xx})(1 - u_{yy}) - u_{xy}^2 = f(x, y), \tag{2.7}$$

with Dirichlet boundary conditions on a unit square, $\Omega = [0, 1] \times [0, 1]$ in xy-plane. They also assume $f(x, y) > 0$ on Ω. The equation (2.7) can be obtained from the general MAE equation (1.3) by setting $a = 1, b = 0,\ c = 1$, right hand side $= f - 1$. Here, $\Theta > 0$, so the equation (2.7) is elliptic.

To discretize (2.7) on a uniform grid Ω_h, the authors consider a discretization of form

$$L^h(u) = \omega^h(f) \tag{2.8}$$

where

$$L^h u = (1 - A_y(a_1)D_{xx}u)(1 - D_{yy}A_x(a_1)u) - (D_{xy}u)^2, \tag{2.9}$$

$$\omega^h f = c(A_x(a_2) + A_y(a_2))f + Sf. \tag{2.10}$$

Here a_1, a_2, and c are constants, and the finite difference operators, D_{xx}, D_{yy}, and D_{xy} are as described in Section 2.1. The averaging operators, $A_x(a)$ and $A_y(a)$, and weighting operator S are given by

$$\begin{aligned} A_x(a)u &= \frac{1}{1+2a}[au_{i-1,j} + u_{i,j} + au_{i+1,j}], \\ A_y(a)u &= \frac{1}{1+2a}[au_{i,j+1} + u_{i,j} + au_{i,j-1}], \\ Sf &= d[f_{i-1,j-1} + f_{i+1,j-1} + f_{i-1,j+1} + f_{i+1,j+1}], \end{aligned} \tag{2.11}$$

where d is a constant. The truncation error of (2.8) can be calculated as

$$\tau = \omega^h f - L^h u = Eh^2 + O(h^4), \tag{2.12}$$

where E is a function of a_1, a_2, c, d and partial derivatives of u. The idea is to obtain a 4^{th} order compact discretization by selecting the parameters a_1, a_2, c, d such that E is as small as possible. Chen suggests four possible choices for a_1, a_2, c, d. Each combination of parameter value leads to a different discretization. Some examples are listed below

1. MV discretization: This is obtained by setting $a_1 = \frac{1}{10}, a_2 = \frac{1}{4}, c = \frac{1}{2}$ and $d = 0$ in (2.9), (2.10) and (2.11).

2. CY discretization: For this discretization, we set $a_1 = \frac{1}{4}, a_2 = 1, c = \frac{1}{2}$ and $d = 0$.

3. Standard discretization: This is obtained by setting $a_1 = 0, a_2 = 0, c = \frac{1}{2}$ and $d = 0$.

4. Conservative discretization: This is obtained by setting $a_1 = \frac{1}{2}, a_2 = 0, c = \frac{1}{2}$ and $d = 0$.

This discretization is claimed to be fourth order accurate. Furthermore, Chen has proposed a multigrid method based on this discretization (2.8). The convergence rate for this method can range from 0.2653 to 0.99. Thus for some cases, the convergence can be very slow.

2.3 Discussion on Viscosity solution and Monotone Discretization

In practice, it may be desirable that the numerical scheme will give the viscosity solution. We recall the definition of viscosity solution [12] for the problem (2.1), (2.2), followed by the definition of a monotone scheme.

Definition (Viscosity Solution): Let $u \in C(\Omega)$ be convex and $f > 0$. The function u, satisfying condition (2.2), is called a viscosity supersolution (subsolution) of (2.1), if for any $\phi \in C^2$ and $x_0 \in \Omega$, such that $(u - \phi)(x) \geq (\leq)(u - \phi)(x_0)$ we have

$$det(D^2\phi(x_0) \leq (\geq) f(x_0).$$

The function u is called a viscosity solution if it is both a *viscosity supersolution* and a *subsolution*.

Definition (Monotone Scheme): A scheme $U_j^{n+1} = B(U^n; j)$, is monotone, if

$$\frac{\partial}{\partial U_i^n} B(U^n; j) \geq 0, \qquad \forall i, j, U^n.$$

This means that the value of U_j^{n+1} cannot decrease as a result of an increase in U_i^n.

In [9] Feng and Nilan present a finite element method to approximate the viscosity solution. Their approach is based on vanishing moment method, which involves approximating a second order boundary value problem by a fourth order boundary value problem.

This requires additional boundary conditions which do not seem to be consistent with the original equation in the limit of regularization parameter tending to zero [17].

Furthermore, we know that solutions of a consistent, monotone finite difference scheme converge uniformly to the unique viscosity solution (*Barles-Souganidis convergence*). In [17], Oberman proposes a monotone finite difference approximation scheme to solve the MAE. They develop a wide stencil scheme to build this monotone method. The main idea is to discretize the direction vector, say η, of the second derivatives and restrict them to the direction, say γ that aligns with the grid. They call this the directional derivative. Let $d\theta$ be the directional resolution of the grid, the distance measure between η and γ. Then $d\theta$, will approach zero as the stencil width is increased. For grid points near the boundary, not all values would be available as required by the wide stencil scheme. In this case, they use quadratic interpolation near boundary to develop a lower accuracy stencil for directional derivative. Obviously, the wide stencil schemes are more complex than narrow stencil schemes.

Our focus on MAE stems from its application in image processing, for which even a second order accurate method is sufficient. Therefore, we consider a simple discretization given in Section 2.1 corresponding to the MAE (2.1).

We will give an overview of multigrid methods in the following chapter before discussing multigrid in particular for Monge-Ampère equations in Chapter 5.

Chapter 3

Multigrid Overview

Multigrid is one of the most efficient methods for solving a wide variety of partial differential equations. The computational work involved per iteration step is proportional to the number of unknowns [5]. Also, the convergence rate for a multigrid method is often independent of the size of the finest grid in multigrid cycle [5]. In contrast, for standard iterative methods the convergence becomes slower as the mesh size is reduced.

Multigrid is an algorithm for solving differential equations using a hierarchy of coarse grids. Problems solved by multigrid methods include general elliptic PDEs, nonlinear and eigenvalue problems, and systems of equations from fluid dynamics [5]. The main idea behind multigrid is to accelerate the convergence of a basic iterative method by solving a coarse problem to reduce the error, from time to time. This is achieved by first smoothing the error of an approximate solution, followed by approximation of the smooth error on a coarse grid, i.e. a grid with substantially fewer grid points. This chapter gives an overview of the multigrid method.

3.1 Multigrid components

We briefly discuss four main components to develop a multigrid method below,

- Smoother - Smoother is a relaxation scheme which reduces high frequency (or oscillatory) errors, leaving the smooth error to be approximated by multigrid cycle.
- Coarse Grid - This is a selection process where we reduce the number of grid points under consideration to define a coarser grid.

- Restriction - This is an operator which downsamples the discrete function from fine to a coarser grid.
- Prolongation - This operator interpolates a correction computed on a coarse grid onto the finer grid.

3.2 Multigrid algorithm

To illustrate the steps involved in multigrid algorithm, consider the one dimensional Poisson equation

$$\begin{aligned} u_{xx} &= f(x), \qquad 0 < x < 1 \\ u(0) &= u(1) = 0. \end{aligned} \tag{3.1}$$

Define a uniform sequence of N grid points in [0,1] denoted by x_i, $i = 1, 2, ..., N$, where $x_1 = 0$ and $x_N = 1$. We refer to this discretized domain as Ω_h, where $h = \frac{1}{N-1}$ is the mesh size. The objective is to approximate the value of the solution u on Ω_h, i.e. approximate $u(x_i) =$, $1 \leq i \leq N$.

Applying a finite difference or a finite element method, the problem, (3.1), can be discretized on Ω_h and represented by

$$A_h u_h = f_h, \tag{3.2}$$

where A_h is the discrete Laplacian matrix. Let u_h^k be the k^{th} approximation on Ω_h. The error associated with u_h^k is composed of high and low frequency components. We know that to approximate an oscillatory function we require much more number of data points, than to effectively capture a smooth function. We apply this idea to a multigrid iteration by dividing the error approximation into two stages. First, we apply a relaxation scheme to reduce the high frequency component of the error on a fine grid. Second, the low frequency or the smooth error is approximated on a coarser grid. Thus found error approximation can be used to improve the approximate solution u_h^k to obtain u_h^{k+1}.

We now discuss a two-grid algorithm to obtain a better approximation u_h^{k+1}. In a two-grid method the problem is defined on a fine grid Ω_h and the smooth component of the error is approximated on a coarse grid, given by Ω_H, where H is the mesh size of the coarse grid.

$$u_h^{k+1} = MultiGrid(A_h, f_h, u_h^k, \alpha_1, \alpha_2, level) \tag{3.3}$$

1. Presmoothing

   - Compute $\tilde{u}_h^k$ by applying α_1 iterations of a relaxation scheme (e.g., Gauss-Seidel, Jacobi, etc) with u_h^k as initial guess:

$$\tilde{u}_h^k = Smoother(u_h^k, A_h, f_h, \alpha_1) \tag{3.4}$$

2. Coarse Grid Correction
   In this step the problem is restricted to and solved on the coarse grid.

   - Compute the residual, r_h^k on the fine grid

$$r_h^k = f_h - A_h \tilde{u}_h^k. \tag{3.5}$$

   - Restrict the residual onto the coarse grid, Ω_H

$$r_H^k = R_h^H r_h^k. \tag{3.6}$$

     Here, R_h^H is the restriction operator. Now the coarse grid problem is the error equation defined on Ω_H, given by

$$A_H e_H^k = r_H^k, \tag{3.7}$$

     and the objective at this level is to approximate e_H^k. Here A_H is the coarse grid matrix.

   - Compute the solution, e_H^k, to (3.7) by employing a direct solver.
   - Interpolate the error correction onto the fine grid, Ω_h

$$e_h^k = I_H^h e_H^k, \tag{3.8}$$

     where I_H^h is the Prolongation operator.

   - Update the solution on the fine grid, Ω_h.

$$u_h^{k,CGC} = \tilde{u}_h^k + e_h^k \tag{3.9}$$

3. Postsmoothing

   - Compute u_h^{k+1} by applying α_2 iterations of the relaxation scheme with $u_h^{k,CGC}$ as initial guess:

$$u_h^{k+1} = Smoother(u_h^{k,CGC}, A_h, f_h, \alpha_2) \tag{3.10}$$

The number of iterations for presmoothing, α_1, in (3.4), is typically a small number, e.g. 1 or 2. In (3.10), the number of sweeps for postsmoothing is usually equal to presmoothing, i.e. $\alpha_2 = \alpha_1$. Furthermore, to compute the coarse grid matrix for equation (3.7), we can use the Galerkin matrix, $A_H = R_h^H A_h I_H^h$. Alternatively, the problem (3.2) can be discretized on the coarse grid, Ω_H, to obtain A_H.

3.3 The Multigrid cycle

For a problem defined on a very fine grid, the two-grid method is not practical since the coarse grid problem is still expensive to approximate. Thus, to reduce the computational complexity, we can recursively apply the idea to restrict the problem (3.7) to an even coarser grid than Ω_H. On the coarsest grid, the problem can be solved for a suitable approximation which can be interpolated through all the intermediate grids to the fine grid Ω_h. Each grid in the cycle is referred to as a *Level*.

Usually, to obtain a good approximate solution of (3.7), it is sufficient to perform the coarse grid correction a few, say β, times. The case when $\beta = 1$, is called a *V-cycle*. In a *V-cycle* the problem is restricted all the way to coarsest grid and then correction is interpolated to the fine grid. Another interesting case is $\beta = 2$, which is referred to as a *W-cycle*. See Figure 3.1 for structure of multigrid cycle for $\beta = 1, 2$ on different grids.

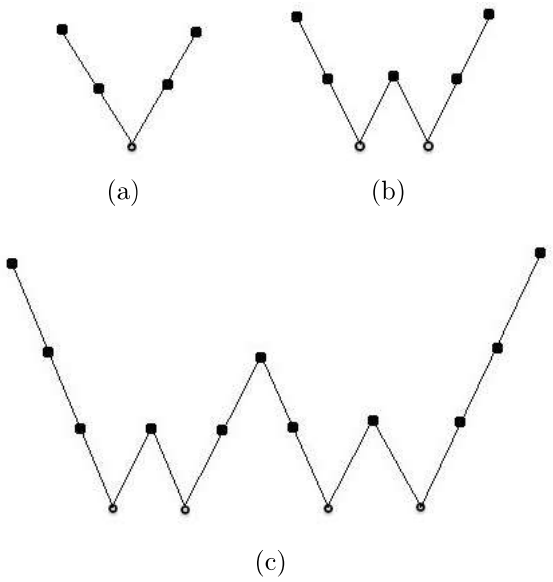


Figure 3.1: Structure of multigrid cycles for different grids and β. (a) Three-grid method with $\beta = 1$. (b) Three-grid method with $\beta = 2$. (c) Four-grid method with $\beta = 2$.

The algorithm for multigrid cycle can be summarised as:

1. Define problem on fine grid and specify number of Levels for multigrid

2. (a) If $Level = 1$,
   Employ a fast solver for the problem (i.e. by a direct solver or many relaxation sweeps)

   (b) else,
   Solve the problem approximately by applying β times the (Level-1)-grid cycles with zero as the initial approximation.
   (i.e. solve recursively with (Level = Level -1) and Go to 2(a))

For more information about multigrid methods, please refer to Trottenberg et al [5] and Brandt [2].

Chapter 4

Multigrid for Monge-Ampère Equation

As highlighted in Chapter 1 and 2, many numerical methods have been proposed to solve the MAEs. However, for the larger problems the computations are expensive and convergence is slow. From Chapter 3, we know that the convergence of multigrid is independent of the grid size. Additionally the computational work for multigrid is proportional to the number of unknowns. Therefore we explore multigrid to devise an efficient and fast method to solve the elliptic MAEs.

In Section 2.2, we described the discretization proposed by Ye Chen [3] for the elliptic MAE. The author also developed a multigrid method to solve the discrete equation (2.8). He uses bilinear interpolation and full weighting. Gauss-Seidel and Jacobi are derived and used as smoothers. For a problem defined on 48×32 grid, the author compares the performances (in terms of computation time) of his multigrid method and the relaxation scheme on a single grid, which comes out to be 3 hours and 64.93 hours respectively. Furthermore, the convergence factor for Gauss-Seidel and Jacobi range from 0.2653 to 0.99 and 0.3425 to 0.99, respectively. The huge variation in the convergence indicates that the method can be slow for some cases. More specifically, for the optimal Gauss-Seidel (unweighted), multigrid with V(1,1) has a convergence factor of 0.2653 while V(2,1) corresponds to 0.3904 for the same data. Similarly, for optimal Jacobi (weighted with $\omega = 0.9$), the convergence factors corresponding to V(1,1) and V(2,1) are 0.3425 and 0.4263, respectively. These results are surprising because we can see that two sweeps of pre-smoothing leads to a slower convergence than the cycle with one sweep.

Our development of multigrid is based on the second order discretization of the non

linear elliptic MAE, as discussed in Section 2.1. The domain under consideration is a square, Ω with N grid points along each coordinate axis and Dirichlet boundary conditions are assumed. A reasonable choice for the initial guess is the solution of $u_{xx} + u_{yy} = \sqrt{2f}$ [16].

4.1 Relaxation Scheme & Smoothing

In this section, we discuss the relaxation scheme which would be used to remove the high frequency error. Consider the discretized MAE derived from (2.6)

$$4(a_1 - u_{i,j})(a_2 - u_{i,j}) - \frac{1}{4}(a_3 - a_4)^2 = h^4 f_{i,j},$$

where

$$\begin{aligned} a_1 &= \frac{1}{2}[u_{i,j+1} + u_{i,j-1}], \\ a_2 &= \frac{1}{2}[u_{i+1,j} + u_{i-1,j}], \\ a_3 &= \frac{1}{2}[u_{i+1,j+1} + u_{i-1,j-1}], \\ a_4 &= \frac{1}{2}[u_{i-1,j+1} + u_{i+1,j-1}]. \end{aligned}$$

It can be rewritten as,

$$4u_{i,j}^2 - 4(a_1 + a_2)u_{i,j} + 4a_1a_2 - \frac{1}{4}(a_3 - a_4)^2 - h^4 f_{i,j} = 0.$$

Solving this quadratic equation for $u_{i,j}$ gives two roots:

$$u_{i,j} = \frac{1}{2}(a_1 + a_2) \pm \frac{1}{2}\sqrt{(a_1 - a_2)^2 + \frac{1}{4}(a_3 - a_4)^2 + h^4 f(i,j)}. \tag{4.1}$$

We select the smaller root to ensure a convex solution [16]. The convexity constraint is necessary for the uniqueness. Otherwise, existence of a solution u implies that -u is also a solution of (2.1) [16]. Considering u^k to be the current approximation, the updated

solution $\tilde{u}^k$ is computed by

$$
\begin{aligned}
a_1 &= \frac{1}{2}[u^k_{i,j+1} + u^k_{i,j-1}];\\
a_2 &= \frac{1}{2}[u^k_{i+1,j} + u^k_{i-1,j}];\\
a_3 &= \frac{1}{2}[u^k_{i+1,j+1} + u^k_{i-1,j-1}];\\
a_4 &= \frac{1}{2}[u^k_{i-1,j+1} + u^k_{i+1,j-1}];\\
\tilde{u}^k_{i,j} &= \frac{1}{2}(a_1 + a_2) - \frac{1}{2}\sqrt{(a_1 - a_2)^2 + \frac{1}{4}(a_3 - a_4)^2 + h^4 f(i,j)}.
\end{aligned} \tag{4.2}
$$

There is no convergence proof for this method. However, in our experiments this scheme converges. We will use this relaxation scheme as the smoother for our multigrid method.

4.2 Multigrid Elements

In this section, we present the important choices with respect to the multigrid elements.

4.2.1 Coarse Grid

Given a fine grid Ω_h, the most commonly used coarsening is to double the mesh size in each direction to obtain a coarse grid, Ω_H, where grid size $H = 2h$. This is called full coarsening. Another option for higher dimensional (2D and above) problems is to double the grid size along any one coordinate axis. This is called semicoarsening. In this paper, we will only consider full coarsening.

4.2.2 Restriction

A restriction operator maps functions, such as residual from Ω_h onto Ω_H. One simple approach is to consider the weighted average of all the neighboring points. This is called full weighting operator. Consider the below scenario of solution values at grid points in Ω_h. For notational convenience, here we represent a solution point on Ω_h, $u^k_h(x_i, y_j)$ by $u^h_{i,j}$

and a point $u_H^k(x_i, y_j)$ on Ω_H by $u_{i,j}^H$.

$$\begin{bmatrix} u_{i-1,j+1}^h & u_{i,j+1}^h & u_{i+1,j+1}^h \\ u_{i-1,j}^h & u_{i,j}^h & u_{i+1,j}^h \\ u_{i-1,j-1}^h & u_{i,j-1}^h & u_{i+1,j-1}^h \end{bmatrix}. \tag{4.3}$$

In this case, the coarse grid point $u_{i,j}^H$ is given by:

$$u_{i,j}^H = \frac{1}{4}\left\{ u_{i,j}^h + \frac{1}{2} \times [u_{i,j-1}^h + u_{i-1,j}^h + u_{i+1,j}^h + u_{i,j+1}^h] \right.$$
$$\left. + \frac{1}{4} \times [u_{i-1,j-1}^h + u_{i+1,j-1}^h + u_{i-1,j+1}^h + u_{i+1,j+1}^h] \right\}.$$

The restriction operator can be written in the stencil form as

$$R_h^H = \frac{1}{4} \begin{bmatrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{2} & 1 & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \end{bmatrix}.$$

This operator is used for all the results in this paper.

4.2.3 Prolongation

The function mapping from Ω_H to Ω_h is called prolongation (also known as interpolation). A commonly used prolongation operator is bilinear interpolation. Consider a neighbourhood of grid points in Ω_H

$$\begin{bmatrix} u_{i-2,j+2}^H & . & u_{i,j+2}^H & . & u_{i+2,j+2}^H \\ . & . & . & . & . \\ u_{i-2,j}^H & . & u_{i,j}^H & . & u_{i+2,j}^H \\ . & . & . & . & . \\ u_{i-2,j-2}^H & . & u_{i,j-2}^H & . & u_{i+2,j-2}^H \end{bmatrix}. \tag{4.4}$$

In this case the fine grid points $u_{i+1,j}^h$ and $u_{i+1,j+1}^h$ are given by

$$u_{i+1,j}^h = \frac{1}{2} \times [u_{i,j}^H + u_{i+2,j}^H],$$
$$u_{i+1,j+1}^h = \frac{1}{4} \times [u_{i,j}^H + u_{i+2,j}^H + u_{i,j+2}^H + u_{i+2,j+2}^H].$$

Bilinear interpolation, in stencil form, can be defined as

$$P_H^h = \begin{bmatrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{2} & 1 & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \end{bmatrix}.$$

4.2.4 Coarse Grid Problem

Since we are working with fully non linear equation, it is challenging to solve the problem exactly even on a coarse grid. Hence we apply the relaxation scheme to solve the problem approximately on the coarse grid. For instance, if the problem is of the form $A(u) = f$, with the initial guess and k^{th} approximation given by u^0 ,u^k respectively, then we can select either of the following as the stopping criterion for the iterative scheme.

$$\left|\left|r^k\right|\right| < Tolerance \times \left|\left|r^0\right|\right| \tag{4.5}$$

or

$$\left|\left|r^k\right|\right| < Tolerance \times ||f||\,. \tag{4.6}$$

Here, r^q denoted the residual corresponding to the q^{th} approximation u^q.

We use the condition from (4.5) as the stopping criterion for all the results in this paper. A reasonable *Tolerance* in this case could be in the range 10^{-4} to 10^{-8}.

4.3 Full Approximation Scheme

The multigrid method described so far uses the correction scheme, wherein the coarse grid solution is an approximate correction to the fine grid. This approach is applicable to linear problems as residual problem is dependent on the linearity of the operator.

Consider the discretized MAE from (2.6). For simplified notations, it can be rewritten as $N_h(u_h) = f_h$ (where $N_h(.)$ is a non linear operator), on Ω_h, with the approximate solution given by u_h^k. The relation between exact and approximate solution is expressed by $u_h = u_h^k + e_h^k$, where e_h^k is the error. Furthermore, the error equation is given by

$$f_h - N_h(u_h^k) = r_h^k, \tag{4.7}$$

where r_h^k is called residual. The equation (4.7) can be rewitten as

$$N_h(u_h) - N_h(u_h^k) = r_h^k \tag{4.8}$$

or equivalently,

$$N_h(u_h^k + e_h^k) - N_h(u_h^k) = r_h^k. \tag{4.9}$$

From (4.9) it does not seem straight forward to approximate e_h^k.

To numerically solve nonlinear problems, we apply a generalized multigrid algorithm called full approximation scheme (FAS). The primary idea behind nonlinear multigrid is the same as the linear case. To begin with, the error of the solution is smoothed so that they can be approximated on a coarser grid, followed by the restriction of problem onto coarser grids, where the residual equation is solved. Subsequently the coarse grid correction is interpolated to the fine grid, where the errors are again smoothed. The only difference lies in the analog of the coarse grid problem. In case of FAS, we work with full approximation of the solution on the coarse grid, rather than the errors, as in the case of linear multigrid.

Instead of working with residual equation on the coarse grid, FAS involves solving the following equation for u_H^k

$$N_H(u_H^k) = R_h^H r_h^k + N_H(R_h^H u_h^k). \tag{4.10}$$

Thus obtained u_H^k is used to compute error on coarse grid, which is interpolated to fine grid to give an improved approximation.

Given u_h^k, the k^{th} approximation to MAE on Ω_h, the stepwise nonlinear FAS algorithm to obtain improved solution u_h^{k+1} is described below

$u_h^{k+1} = FAS(u_h^k, N_h, f_h, \alpha_1, \alpha_2, Level)$

If Level = 1
- Apply Gauss-Seidel to solve the problem within *Tolerance*

else,

1. Presmoothing
   Compute $\tilde{u}_h^k$ by applying α_1 iterations of the relaxation scheme (4.2) to u_h^k

   $$\tilde{u}_h^k = Smoother(u_h^k, N_h, f_h, \alpha_1).$$

2. Coarse Grid Correction

(a) Compute the residual

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

(b) Restrict the residual

$$\tilde{r}_H^k = R_h^H \tilde{r}_h^k.$$

(c) Restrict $\tilde{u}_h^k$

$$\tilde{u}_H^k = R_h^H \tilde{u}_h^k.$$

(d) Compute the right hand side

$$f_H = \tilde{r}_H^k + N_H(\tilde{u}_H^k).$$

(e) Compute an approximate solution $\hat{v}_H^k$ of the coarse grid equation on Ω_H

$$N_H(\hat{v}_H^k) = f_H.$$

For this solve (4.11) approximately by using $\tilde{u}_H^k$ as the initial approximation

$$\hat{v}_H^k = FAS(\tilde{u}_H^k, N_H, f_H, \alpha_1, \alpha_2, Level - 1).$$

(f) Compute coarse error

$$\hat{e}_H^k = \hat{v}_H^k - \tilde{u}_H^k.$$

(g) Interpolate the coarse error

$$\hat{e}_h^k = P_H^h \hat{e}_H^k.$$

(h) Update the solution on Ω_h

$$u_h^{k,CGC} = \tilde{u}_h^k + \hat{e}_h^k.$$

3. Postsmoothing
Compute u_h^{k+1} by applying α_2 iterations of the relaxation scheme (4.2) to $u_h^{k,CGC}$

$$u_h^{k+1} = Smoother(u_h^{k,CGC}, N_h, f_h, \alpha_2).$$

Chapter 5

Numerical Results

In this chapter, we apply the proposed multigrid method for MAE on five examples with smooth to moderately singular solutions. Computations were performed on a Mac desktop with 2.8GHz Intel Core 2 Duo processor and 4GB memory, using MATLAB running in Mac OS X.6.

In particular, we are interested in comparing the performance of the multigrid solver with the relaxation scheme proposed by Benamou, Froese and Oberman (4.2). The comparison is based on two measures, the number of iterations and the computation (CPU) time.

All the examples that we consider here, were used by Oberman in [16]. Dean and Glowinski [8] give results for example 1, 3 and 5. Their method converges for the first two and diverges for the last.

For all the example problems, the initial approximation is the solution of $u_{xx} + u_{yy} = \sqrt{2f}$. We use full coarsening to select the coarse grid points, while full weighting and bilinear interpolation are used as restriction and prolongation operators respectively. Also, the coarsest grid on which the problem is solved approximately is 5×5.

5.1 Example 1

Consider a problem [16, 8] given by

$$detD^2u = (1 + |x|^2)e^{|x|^2} \quad in\ \Omega, \tag{5.1}$$

$$u = g \text{ on } \partial\Omega.$$

Here Ω is a square the domain $[-1,1]\times[-1,1]$ with $\partial\Omega$ representing the boundary. The function g is given by

$$g(x) = \begin{cases} e^{\frac{1}{2}(1+x_1^2)} & on\ \{x|-1<x_1<1, x_2=1\}\ and\ \{x|-1<x_1<1, x_2=-1\} \\ e^{\frac{1}{2}(1+x_2^2)} & on\ \{x|x_1=1, -1<x_2<1\}\ and\ \{x|x_1=-1, -1<x_2<1\}. \end{cases}$$

An exact solution of this problem is given by, $u(x_1,x_2) = e^{\frac{1}{2}|x|^2}$.

This is a simple test example with a smooth solution. Surface plots for the approximated solution and f are shown in Figure 5.1(a), 5.1(b), respectively. From Table 5.1, we observe that the FAS solver converges in 5 to 6 iterations (or V-cycles) for all grid sizes it was tested on. However, the computation time per iteration increases, as the grid is refined. This is expected because finer the grid, deeper the V cycle, in other words more number of intermediate grids between fine and the coarsest. On the contrary, the number of iterations corresponding to the relaxation scheme increases (approx) four times as the grid size is halved. In Figure 5.1(c), 5.1(d) we plot the number of iterations and computation time against the number of grid points for both the methods.

N	Iterations		CPU Time (sec)	
	Relaxation Scheme	FAS	Relaxation Scheme	FAS
17	300	5	0.460	0.064
33	1176	6	7.086	0.208
65	4647	6	114.858	0.764
129	20692	6	2282.684	3.443
257	–	6	–	19.196
513	–	6	–	176.194

Table 5.1: Example 1- Number of iterations and computation times for relaxation method and FAS.

Note: '–' in the table denotes that the problem took more than 5 hours to converge to the exact solution. So the corresponding iterations and CPU time were too large.

5.2 Example 2

We again consider the same problem from Example 1, and study the effects of adding high frequency sinusoidal noise to the right-hand-side, i.e., *f(x,y)* and the boundary conditions.

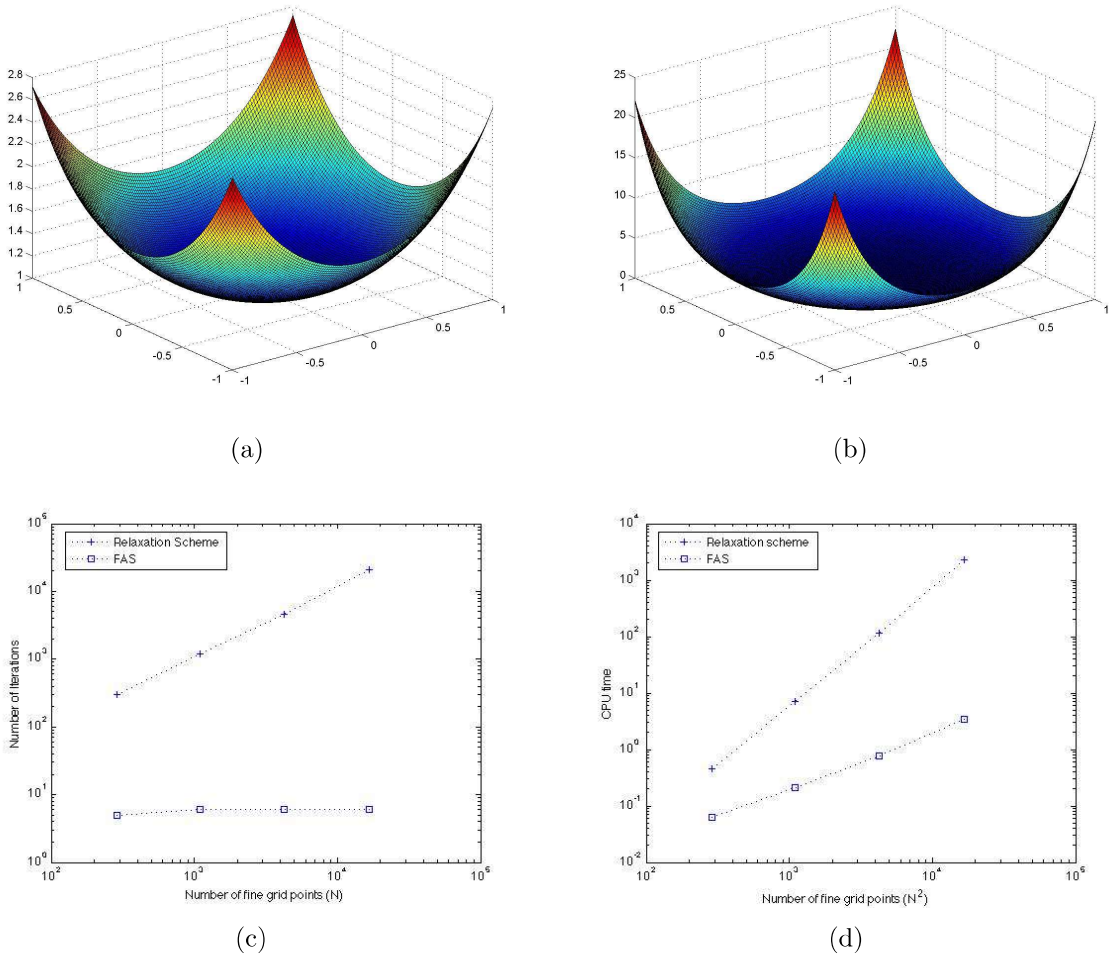


Figure 5.1: Example 1 (a) Surface plot of solution on 129 x 129 grid. (b) Surface plot of f(x,y) on 129 x 129 grid. (c) Number of grid points versus number of iterations for the Relaxation Scheme and FAS. (d) Number of grid points versus CPU time for the Relaxation Scheme and FAS.

Here the right hand side is given by

$$f(x) = (1 + |x|^2)e^{|x|^2} + \sin(40\pi x_1 + 40\pi x_2) \tag{5.2}$$

and the boundary conditions is:

$$g(x) = \begin{cases} e^{\frac{1}{2}(1+x_1^2)} + \mu \sin(40\pi x_1 + 40\pi) & on\ \{x| -1 < x_1 < 1, x_2 = 1\} \\ e^{\frac{1}{2}(1+x_1^2)} + \mu \sin(40\pi x_1 - 40\pi) & and\ \{x| -1 < x_1 < 1, x_2 = -1\} \\ e^{\frac{1}{2}(1+x_2^2)} + \mu \sin(40\pi + 40\pi x_2) & on\ \{x|x_1 = 1, -1 < x_2 < 1\} \\ e^{\frac{1}{2}(1+x_2^2)} + \mu \sin(-40\pi + 40\pi x_2) & and\ \{x|x_1 = -1, -1 < x_2 < 1\}. \end{cases}$$

The noise added to the boundary conditions, μ, is dependent on the mesh size of the problem. As the mesh is refined, a smaller μ is selected so that the scale of noise is not very pronounced on the boundary. For instance, on a 17×17 grid, we consider $\mu = 0.01$ and for 65×65, value of choice is $\mu = 0.001$. For the function f, we add the same noise with $\mu = 1$. In Figure 5.2(c) we can see the scale of the error defined on 33×33.

Figure 5.2(b) illustrates the solution which is convex, except at the boundaries due to the added noise. Although the input data is not smooth, see Figure 5.2(a) for plot of f. Furthermore, the data from the Table 5.2 shows that the noise has no noticeable effect on the convergence of either method. Also, for all the problem sizes, FAS is the faster converging method of the two.

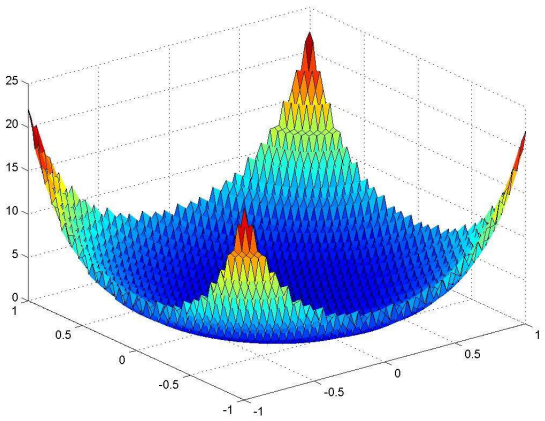
N	Iterations		CPU Time (sec)	
	Relaxation scheme	FAS	Relaxation Scheme	FAS
17	328	5	0.495	0.084
33	1298	6	7.881	0.198
65	5145	6	127.691	0.766
129	19863	6	2187.932	3.458
257	–	6	–	30.122

Table 5.2: Example 2 - Number of iterations and computation times for the Relaxation scheme and FAS.

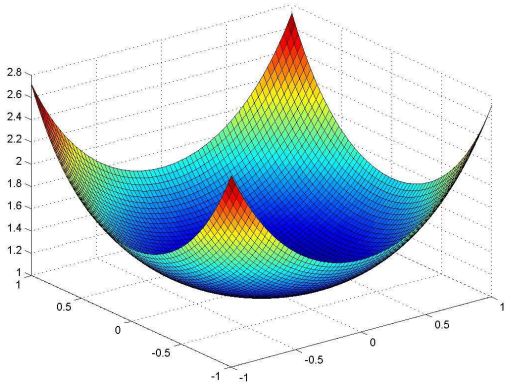
5.3 Example 3

Consider the following exact solution on the square $[0, 1] \times [0, 1]$.

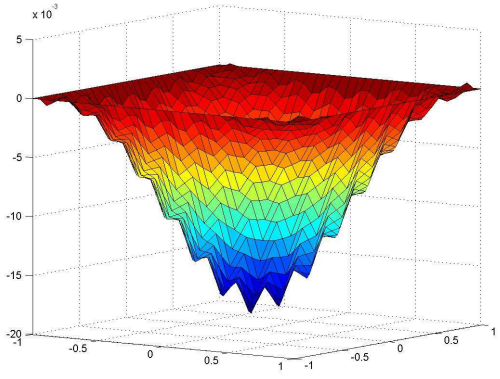
$$u^{exact}(x, y) = \frac{2\sqrt{2}}{3}(x^2 + y^2)^{3/4}, \qquad f(x, y) = \frac{1}{\sqrt{x^2 + y^2}}. \tag{5.3}$$



(a)



(b)



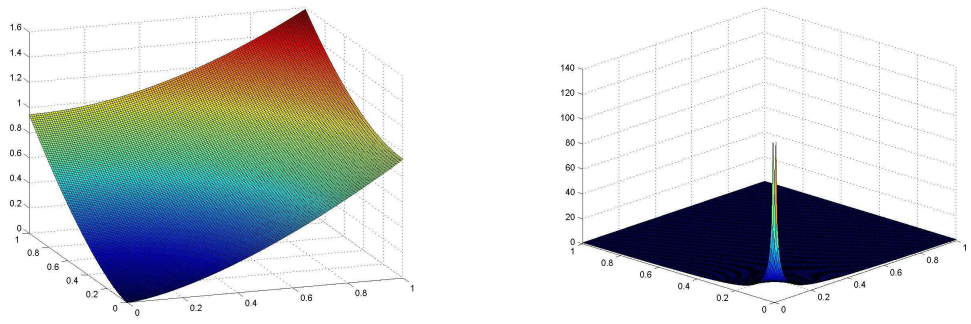
(c)

Figure 5.2: Example 2 (a) Surface plot of f(x,y) on 65 x 65 grid. (b) Surface plot of solution on 65 x 65 grid. (c) Error on 33 x 33 grid.

This is a mildly singular problem as the right hand side function, f is unbounded at the boundary point (0,0) and near the origin it can be large; see Figure 5.3(b). However, according to [8], this problem has enough regularity, in terms of the solution being in $H^2(\Omega)$ space, for their algorithm to be applied to obtain an approximation. Solution obtained by application of our solver is smooth, as illustrated in Figure 5.3(a). Moreover, as we can see from Table 5.3, both the methods converge to the exact solution. FAS converges in about 5 to 6 cycles and the iterations for the relaxation scheme increasing as the grid is refined. Comparing the computation times, the FAS is significantly faster than the iterative method for all problem sizes considered.

N	Iterations		CPU Time (sec)	
	Relaxation scheme	FAS	Relaxation scheme	FAS
17	364	6	0.539	0.053
33	1451	6	8.784	0.141
65	5778	7	143.375	0.599
129	22996	7	2545.834	2.866
257	–	7	–	17.595
513	–	7	–	186.265

Table 5.3: Example 3 - Number of iterations and computation times for the relaxation scheme and FAS.



(a) (b)

Figure 5.3: Example 3 (a) Surface plot of solution on 129 x 129 grid; (b) Surface plot of f(x,y) on 129 x 129 grid.

5.4 Example 4

Next we study the following solution defined on $\Omega = [0,1] \times [0,1]$,

$$u(x,y) = max\left\{\frac{(x-0.5)^2 + (y-0.5)^2}{2}, 0.08\right\} \tag{5.4}$$

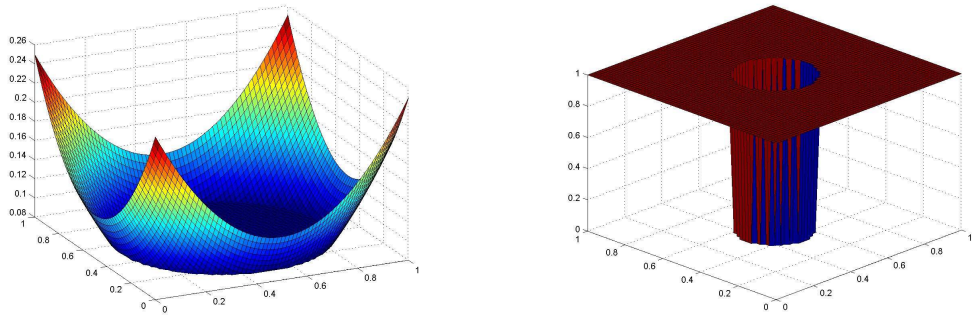
$$f(x,y) = \begin{cases} 1 & (x-0.5)^2 + (y-0.5)^2 > 0.16^2 \\ 0 & otherwise. \end{cases}$$

Here f(x,y) is discontinuous on a circle with radius 0.16, and centred at $(0.5, 0.5)$; see Figure 5.4(b). Moreover, u_{xy} does not exist near the boundary of this circle in Ω. In (5.4), u is a solution in the sense that it satisfies the MAE at all points inside as well as outside the circular region. The solution u is flat, with grid value 0.08, in the circular region where f is zero. Figure 5.4(a) illustrates the surface plot of solution u.

As observed from Table 5.4, the convergence of FAS for this example is slower. The number of V-cycles increases everytime the grid size is reduced by half, while for all the preceding examples the iterations were in the range 5-7. Recall the underlying idea behind multigrid; a smooth solution can be well approximated using fewer points than an oscillatory one. In this example it is evident from the results, that the non-smooth solution is not appromimated well on the coarsest grid (5×5), hence more iterations are required to reach a good approximation. The convergence can be improved by performing one-sided interpolation near the boundary of the circle and linear interpolation on the remaining domain. Inspite of a higher number of iterations than usual, the computation time for FAS is still considerably lower than the relaxation scheme.

N	Iterations		CPU Time (sec)	
	(Iter Method)	(FAS)	(Iter Method)	(FAS)
17	326	6	0.488	0.054
33	1297	9	7.864	0.204
65	5216	16	129.358	1.343
129	–	20	–	8.073
257	–	28	–	69.182

Table 5.4: Example 4 - Number of iterations and computation times for the relaxation scheme and FAS.



(a) (b)

Figure 5.4: Example 4 (a) Surface plot of solution on 65 x 65 grid; (b) Surface plot of f(x,y) on 65 x 65 grid.

5.5 Example 5

Consider another problem with exact solution defined on $[0,1] \times [0,1]$ given by,

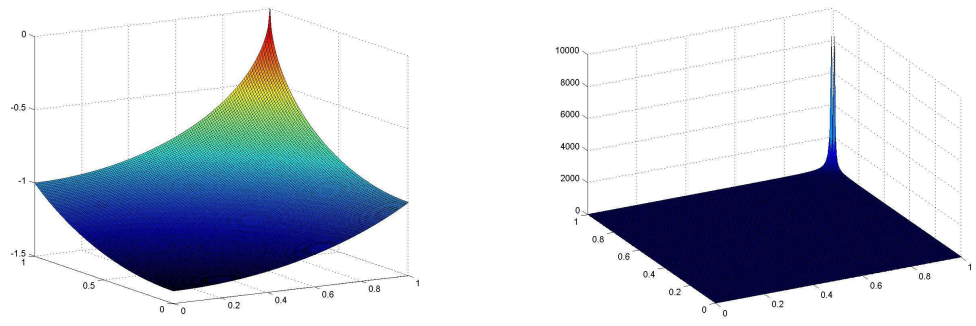
$$u^{exact}(x,y) = -\sqrt{R - x^2 - y^2},$$

$$f = \frac{2}{(R - x^2 - y^2)^2}.$$

For this problem, the case $R = 2$ is interesting, because then the gradient of f is unbounded at (1,1). Furthermore, f is unbounded at (1,1), which leads to singularity. See Figure 5.5(a), 5.5(b). This example was also used for testing by Dean and Glowinski [8], and their method is known to have diverged. It could possibly be due to the singularity that the method proposed is unstable for this case. (Their method converges for $R > 2 + 10^{-1}$). Table 5.5, shows that the proposed FAS and the iterative scheme from [16] converge to the exact solution, with FAS being the faster method of the two.

N	Iterations		CPU Time (sec)	
	Relaxation scheme	FAS)	Relaxation scheme	FAS
17	251	5	0.378	0.053
33	865	6	5.242	0.145
65	2900	6	72.045	0.528
129	–	6	–	2.436
257	–	7	–	17.511
513	–	8	–	212.376

Table 5.5: Example 5 - Number of iterations and computation times for the relaxation scheme and FAS.



(a) (b)

Figure 5.5: Example 5 (a) Surface plot of solution on 129 x 129 grid. (b) Surface plot of f(x,y) on 129 x 129 grid.

Chapter 6

Conclusions

We have presented a full approximation scheme (FAS) for solving nonlinear elliptic Monge-Ampère equation (2.1). We use a second order discretization for the equation. Since solving a nonlinear problem exactly on a coarse grid can be complicated, we solve for an approximation by applying the relaxation scheme (4.2). Using several test problems we are able to demonstrate how our multigrid algorithm compares to the existing relaxation scheme for MAE. The numerical results indicate that FAS outperforms (4.2) consistently, in terms of computation time, when the exact solution is smooth. Moreover, even when the exact solution is moderately non-smooth, FAS converges faster than the relaxation scheme. However, in the latter case, FAS's performance is slower when compared to the results produced by its application on examples with smooth solutions.

References

[1] A.D. Aleksandrov. Certain estimates for the Dirichlet problem. *Sov. Math. Dokl.*, 1:1151–1154, 1961.

[2] Achi Brandt. *Multigrid Techniques: 1984 Guide with Applications to Fluid Dynamics.* GMD, 1984.

[3] Ye Chen. *Efficient and Robust Solvers for Monge-Ampère Equations.* PhD thesis, Clarkson University, 2009.

[4] S.Y. Cheng and S.T. Yau. On the regularity of the Monge-Ampère equation $\det(\partial^2 u/\partial x_i \partial x_j) = f(x, u)$. *Commun. Pure. Appl. Math*, 30(1):41–68, 1977.

[5] U. Trottenberg Cornelis W. Oosterlee and Anton Schuller. *Multigrid.* Academic Press, 2000.

[6] Roland Glowinski Edward J. Dean. An augmented Lagrangian approach to the numerical solution of the Dirichlet problem for the elliptic Monge-Ampère equation in two dimensions. *Electron. Trans. Numer. Anal.*, 22:71–96(electronic), 2006.

[7] Roland Glowinski Edward J. Dean. On the numerical solution of the elliptic Monge-Ampère equation in dimension two: a least-squares approach. In *Partial differential equations, volume 16 of Comput. Methods Appl. Sci.*, pages 43–63. Springer, Dordrecht, 2008, 2008.

[8] R. Glowinski E.J. Dean. Numerical methods for fully nonlinear elliptic equations of the Monge-Ampère type. *Computer methods in applied mechanics and engineering*, 195(13-16):1344–1386, 2006.

[9] Xiaobing Feng and Michael Neilan. Mixed finite element methods for the fully nonlinear Monge-Ampère equation based on the vanishing moment method. *SIAM J. Numer. Anal.*, 47(2):1226–1250, 2009.

[10] Xiaobing Feng and Michael Neilan. Analysis of Galerkin methods for the fully nonlinear Monge-Ampère equation. *J Sci. Comput*, 47:303–327, 2011.

[11] D. Gilbarg and N.S. Trudinger. *Elliptic partial differential equations of second order.* Springer, 2001.

[12] C.E. Gutièrrez. The Monge-Ampère equation, progress in nonlinear differential equations and their applications. *Birkhauser Boston Inc.*, 44, 2001.

[13] Steven Haker and Allen Tannenbaum. Optimal mass transport and image registration. In *Proceedings of the IEEE Workshop on Variational and Level Set Methods*, 2001.

[14] Kazufumi Ito. On fluid mechanics formulation of Monge-Kantorovich mass transfer problem. Technical report, Center for Research in Scientific Computation, North Carolina State University, 2007.

[15] Y. Brenier J.D. Benamou and K. Guittet. The Monge-Kantorovitch mass transfer and its computational fluid mechanics formulation. *Int. J. Numer. Meth. Fluids*, 00:1–6, 2000.

[16] Brittany D. Froese Jean-David Benamou and Adam M. Oberman. Two numerical methods for the elliptic Monge-Ampère Equation. *ESAIM: Mathematical Modelling and Numerical Analysis*, pages 737–758, 2010.

[17] A.M. Oberman. Wide stencil finite difference schemes for the elliptic Monge-Ampère equations and functions of the eigenvalues of the hessian. *Discrete and Continuous Dynamical Systems Series B*, 10:221–238, 2008.