Fast Iterative Methods for Variational Models in Image Segmentation

by

Noor Badshah

THE UNIVERSITY
of LIVERPOOL

Thesis submitted in accordance with the requirements
of the University of Liverpool for the
degree of Doctor in Philosophy.

Contents

Acknowledgment

First of all I would like to thank my supervisor Prof. Ke Chen and my second supervisors Dr. David Edward Hodgkinson and Dr. David Mark Lewis for their continuous support throughout my PhD and for providing me opportunity to work on this project. I would like to thank my colleagues Dr. Martyn Hughes, Dr. Joseph Savage, Carlos Brito, and Noppadol Chumchob for their helpful suggestion and useful feedback.

I am very thankful to N. W. F. P University of Engineering and Technology Peshawar (Pakistan) for funding my PhD.

I am very thankful to my lovely wife S. Noor for her support throughout my PhD and for taking care of our children in my absence. I am also very thankful to my parents (Mr and Mrs Gul Badshah) and my family (brothers and sisters) for their prayers and support.

Abstract

Image segmentation is an important branch of computer vision. It aims at extracting meaningful objects lying in images either by dividing images into contiguous semantic regions, or by extracting one or more specific objects in images such as left kidney in CT image. The image segmentation task is, in general, very difficult to achieve since natural images are diverse and complex, and the way we perceive them varies according to individuals.

This thesis proposes four new algorithms for solving variational image segmentation models based on looking for piecewise smooth homogeneous regions and the active contours. The active contours model is more and more used in image segmentation because it relies on solid mathematical properties and its numerical implementation uses the effective level set method to track evolving contours.

The first algorithm in this thesis uses the multigrid method for solving partial differential equations (PDE) which arise from the minimization of the active contour without edges (2-phase image segmentation) of Chan and Vese [35]. Despite of great theoretical properties, the active contours model suffers from the existence of local minima which requires the critical initial guess to get satisfactory results [25, 15]. With our proposed algorithm we can reach the global minimizer. The multigrid method is faster in convergence than uni-level methods like semi implicit (SI) and additive operator splitting (AOS) methods.

The second algorithm in this thesis uses the multigrid method for PDEs which arise from minimization of the multiphase image segmentation model by Vese and Chan [120], which is the extension of the 2-phase image segmentation multigrid method of Noor and Chen [8]. We also develop a new smoother which improves the multigrid method, where standard smoothers can not lead to an efficient multigrid. The multiphase segmentation model depends on an initial guess. To overcome the dependence, we propose a hierarchical 2-phase multigrid for multiphase segmentation.

The third algorithm is an optimization based multilevel method for the 2-phase image segmentation model. This method is used to solve the minimization functional other than solving the PDE arising from minimization. This method allows us to use a very small regularization parameter β used in Chapter 4, yielding improvement in convergence speed in terms of CPU time.

Finally, we propose a new model for selective image segmentation under geometrical conditions such as a set of points and develop an AOS algorithm. This model is used to detect special features in an image, which is a necessary task in almost all medical applications. It allows us to detect objects in a noisy image and speed up the convergence as well.

Overall we are concerned with effective segmentation models, numerical realizations and fast algorithms for image segmentation.

List of Figures

List of Tables

Publications and Presentation

Publications:

- Noor Badshah and Ke Chen, Multigrid method for the Chan-Vese model in variational segmentation, Communication in Computational Physics, 4(2):294–316, 2008.
- Noor Badshah and Ke Chen, On two multigrid algorithms for modeling for variational multiphase image segmentation, IEEE Transactions on Image Processing, 2008, to appear.
- Noor Badshah and Ke Chen, An optimization-based multilevel for variational image segmentation models, Electronic Transaction on Numerical Analysis (ETNA), Submitted, 2008.
- Noor Badshah and Ke Chen, Image segmentation under geometrical conditions using Chan-Vese model, In preparation.

Presentation:

Noor Badshah and Ke Chen, Multigrid method for 2-phase image segmentation models, 22nd Biennial Conference on Numerical Analysis, NA'07 Dundee, Scotland (UK), 2007.

Chapter 1

Introduction

Computer vision is a branch of artificial intelligence that aims at giving vision to machines, which means to develop mathematical models, algorithms and technologies to build a machine with vision capabilities as advanced as human eyesight at least. More mathematically speaking, the purpose of computer vision is to process images acquired with cameras to produce a mathematical representation of the semantic objects in the world.

1.1 Computer Vision and Image Segmentation

Computer vision is divided into image processing, pattern recognition, statistical learning etc., whose objectives are as varied as detection and recognition of objects in images, registration of different views of the same object with different sources, tracking of objects in videos and so on. In this thesis we mainly look into a specific branch of image processing called *Image Segmentation.*

Image segmentation is one part of the general task of computer vision. A ‘common sense’ definition would describe segmentation as the translation of an image, from an array of grey levels, to a symbolic description, for example as a number of well defined regions. For a given image z, the segmentation is the seemingly simple task of separating the foreground from the background. An alternative way to define segmentation is feature based, where the regions of interest in the image are the projections of the ‘objects’ in the scene and then the task of segmentation is to identify and locate these objects, regardless of the indeterminacy due to image acquisition, lighting and so on.

In the last two decades, several mathematical models have been developed to achieve image segmentation. The recent promising models to solve the image segmentation problem are based on variational approaches and Partial Differential Equations (PDE). These models benefit from well-founded mathematical theories that allow us to analyze, understand, improve the existing methods and to work in a continuous setting which makes the proposed models independent of the grid of digital images. This thesis focus on variational image segmentation and active-contour models and algorithms, which share the common feature that they define optimal segmentation as a minimizer of an objective function that generally depends on the given image and the characteristics that are used to identify the different segmented regions. The Euler-Lagrange equation of these models is often be described using a parabolic partial differential equation, which is iterated in time until it reaches steady state.

A contour is introduced into the image to locate boundaries of features and is evolved until steady state thereby dividing the image into regions. A very powerful and popular method for representing the contour is the level-set method originally developed by Osher and Sethian [92], which represents the contour implicitly as a particular (usually the zero) level of a level-set function. The main advantage of this representation is that topological changes, such as merging and pinching off of contours can be captured naturally through smooth changes to the level-set function, because level set is defined in a higher dimension.

In this thesis, we focus mainly on region-based (rather than edge-based) segmentation models. A prototypical example, and the primary one we discuss in this thesis, is the Chan-Vese (CV) "Active Contour Without Edges" model [35], which seeks the desired segmentation as the best piecewise constant approximation to a given image. The CV model can be interpreted as a level-set implementation of the piecewise-constant special case of the more general Mumford and Shah segmentation model [88].

Due to its simplicity and robustness, the CV model has become quite popular and has been adopted in many applications. As a result, a number of generalizations have been developed to improve both its applicability and efficiency. A natural generalization is to multi-channel images. Initially, a vector valued method [32] was used with an application in texture segmentation [102]. Further extensions include object tracking in video sequences in the presence of clutter, registration of images to identify key objects, and color segmentation that can identify an object in an image with an arbitrary combination of colors [115].

Another direction of generalization to the basic Chan-Vese model is to multiphase models, which allow the segmentation of the image into more than two regions. The multiphase method of Vese and Chan [120] only needs $log_2 n$ level-set functions to represent n regions, without any need to avoid overlaping and uncovered regions, drastically improving the efficiency. More recently, J. Lie et al [79] and G. Chung et al [44] have developed novel level-set methods that use only one level-set function to represent an arbitrary number of regions. Another variational model for image segmentation is given by F. Liu et al in [80]. Liu et al proposed an adaptive 2-phase level set image segmentation algorithm to improve the CV model by introduction of a multiplicative gain field. The resulting model is adaptive to intensity inhomogeneity, tends to obtain the actual boundaries of the objects. In [130] Y. Zhang proposed an improved algorithm, where the key functions in the piecewise smooth Mumford and Shah model are replaced by updating the level set function based on an artificial image that is composed of the diffused image and the original image. In [50], X. Du et al proposed a new model where the energy functional contains only two terms: the combined length of the segmentation curves and the high frequency components in regions excluding the object boundaries. Their functional depends on only one variable so we need to solve one PDE to minimize the energy functional. This model can detect both staircase and roof edges in the image. Similar work can be found in [18, 77, 82, 93, 125].

In this thesis the main issue discussed is to improve the computational efficiency of the variational models. The typical approach of gradient flow (i.e., marching the Euler-Lagrange PDE to steady state) usually takes a long time to converge using explicit method, implicit method or additive operator splitting method [127]. In this thesis we develop the multigrid method to solve this PDE using special type of smoother (for both two-phase and multiphase segmentation). Other related work can be found in [95, 94]. Another approach to achieve fast convergence is to treat the models as a discrete optimization problem whose solution is the association of each pixel to a particular region. B. Song et al [110] proposed a direct optimization algorithm, which has the surprising property that for noiseless two-phase images the optimal solution can be provably obtained with only one sweep over the pixels. More segmentation algorithms can be found in [52, 57].

The multigrid method [14, 39, 68, 118, 128] is one of the most powerful numerical methods for solving linear and non-linear elliptic problems. The multigrid method based on the recursive application of error smoothing and coarse grid correction has been demonstrated to be efficient solvers for a wide range of PDEs, although the method is known to be less robust for either case with highly discontinuous coefficients [124]. More references to multigrid methods for different problems in image processing problems are [1, 8, 9, 17, 29, 40, 43, 94, 95, 105].

1.2 Thesis Outline

Chapter 2:

This chapter covers some basic background material such as:

- Definition and explanation of Bounded Variation (BV), Total Variation (TV) and Level set method.
- Some traditional iterative methods for solving system of equations.
- Time marching schemes for solving parabolic partial differential equations and also their stability.
- The Multigrid method implementing on Poisson equation. Smoothing analysis is given for Poisson equation.

Chapter 3:

In this chapter we give literature review of image segmentation.

- We give brief discussion of active contour model without edges by Chan-Vese in this chapter.
- We also discuss some existing methods used for solving partial differential equation arisen from minimization of the Chan-Vese model.

Chapter 4:

In this chapter we describe Chan-Vese model and give details of the existing methods.

- We give details of some smoothers (local and global smoothers).
- We develop multigrid method for solving partial differential equations arisen from minimization of Chan-Vese model.
- We describe smoothing analysis of the smoothers.
- We compare the results obtained from multigrid method with the results obtained from existed methods like Semi Implicit method (SI), Additive Operator Splitting method (AOS) etc.

Chapter 5:

In this chapter we propose an algorithm which uses multigrid for multiphase image segmentation.

- Two phase segmentation model extend to multiphase segmentation model particularly 4-phases (CV4).
- Smoothers discussed in chapter 4 will be used for CV4.
- We propose another smoother in this chapter for multiphase image segmentation model CV4.
- We give the local Fourier analysis of the smoothers.
- We implement our multigrid idea in hierarchical way to extend it from 2 phase to multiphase image segmentation in this chapter.
- We compare the results from different methods discussed in this chapter.

Chapter 6:

In this chapter we present a new optimization based multilevel method for Chan-Vese model.

- We also use this method to the Chan et al model for computing global minima.
- We compare the results from both models and present experimental results form both models.

Chapter 7:

In this chapter we describe an existed model and propose more reliable model for feature selection (image segmentation under geometrical conditions).

- We propose a new model for feature selection.
- We compare the results from both models.

Chapter 8:

In this chapter we give conclusion and discuss some future work.

Chapter 2

Mathematical Preliminaries

In this chapter we describe some useful material to be used in later chapters. Some of the references for this chapter are [7, 14, 39, 56, 68, 92, 98, 106, 122, 128].

This chapter is organized in the following way. Sections 2.1 and 2.2 give some basic definitions, examples and some important theorems. In Section 2.3 we define total variation (TV) and bounded variation (BV) and explain with some examples. Section 2.4 gives details of the level set method and give the numerical implementation of the level set method. Section 2.5 gives some idea of ill posed problems and discuss some regularization techniques. Section 2.6 gives details of some basic iterative methods for solving system of linear equations, and discuss their convergence. Section 2.7 describes some numerical methods for solving parabolic and elliptic PDEs and their stability. We end this chapter with explaining the multigrid method.

2.1 Normed Spaces

Definition 2.1.1 (Seminorm) *A* **vector seminorm** *on a vector space S, is a real valued functional $\|\cdot\|$ such that*

1. $\|\mathbf{x}\| \geqslant 0$ *for all* $\mathbf{x} \in S$,

2. $\|\alpha \mathbf{x}\| = |\alpha|\|\mathbf{x}\|$ *for all* $\alpha \in \mathbb{R}$ *and* $\mathbf{x} \in S$,

3. $\|\mathbf{x}+\mathbf{y}\| \leqslant \|\mathbf{x}\| + \|\mathbf{y}\|$ *for all* $\mathbf{x}, \mathbf{y} \in S$.

A **norm** *is a seminorm if the following additional condition holds:*
$\|\mathbf{x}\| = 0$ *if and only if* $\mathbf{x} = \mathbf{0}$

Examples:

- All norms are seminorms.
- The trivial seminorm, is $\|\mathbf{x}\| = 0 \ \forall \mathbf{x} \in S$.
- The absolute value is a norm on the set of real numbers $\mathbb{R}$.
- **Euclidean norm:**
  Let $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n$ then
  $$\|\mathbf{x}\| = \sqrt{x_1^2 + x_2^2 + \ldots + x_n^2}.$$
  This gives the ordinary distance from the origin to the point $\mathbf{x}$.

- **p-norm:**
  For real number $p \geqslant 1$ and $\mathbf{x} \in \mathbb{R}^n$,

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}}.$$

  Clearly, for $p = 1$ this norm is called 1-norm and for $p = 2$ this is Euclidean norm.

- **Infinity norm/ Maximum norm:**
  For $\mathbf{x} \in \mathbb{R}^n$, $\|\mathbf{x}\|_\infty = \max(|x_1|, |x_2|, \ldots, |x_n|)$.

- **L^p-norm**
  For functions f defined in a domain Ω and $1 \leqslant p < \infty$ then

$$\|f\|_{p,S} = \left(\int_\Omega |f(x)|^p dx \right)^{\frac{1}{p}}.$$

- Total Variation (TV) norm will be discussed later in this chapter.

Definition 2.1.2 (Normed Space) *A vector space S possessing a norm $\|.\|_S$.*

Definition 2.1.3 *A sequence of elements x_n in a normed vector space S is a Cauchy sequence if and only if $\forall \epsilon > 0$ there exist $N \in \mathbb{N}$ such that*

$$\|x_m - x_n\| < \epsilon, \qquad \forall m, n > N.$$

Definition 2.1.4 *A real or complex Banach space S is a complete normed vector space S over the real or complex numbers i.e. with a norm $\|\cdot\|$ such that every Cauchy sequence x_n in S has a limit in S.*

Examples:

- $\mathbb{Q} \subset \mathbb{R}$ is not complete for $\|x\| = |x|$ since there are Cauchy sequences of rational numbers having no limit in $\mathbb{Q}$. Take for instance the sequence $\{x_n : n \in \mathbb{N}\}$ in $\mathbb{Q}$ such that $x_1 = 1$ and $x_{n+1} = \frac{x_n}{2} + \frac{1}{x_n}$ which converges to the irrational $\sqrt{2}$ when $n \to \infty$.

- The set of real numbers $\mathbb{R}$ is complete for $\|x\| = |x|$.

- $\mathbb{R}^n$ is a complete normed space which can be constructed using Cauchy sequences of rational numbers. It is supplied with the norm

$$\|\mathbf{x}\| = \left(\sum_{i=1}^{n} |x_i|^2 \right)^{1/2}.$$

- $C([a, b], \mathbb{R})$, the space of all continuous functions $f : [a, b] \to \mathbb{R}$ is a Banach space if we define the norm of such functions as

$$\|f\| = \sup\{|f(x)| : x \in [a, b]\},$$

  which is known as supremum norm. It is a norm since all continuous functions on a compact interval are bounded.

Definition 2.1.5 (Inner Product) *An inner product on the vector space S is a functional $\langle\cdot,\cdot\rangle_S$ on $S \times S$ which satisfies*

1. $\langle u,u\rangle \geqslant 0$ *for all* $u \in S$.
2. $\langle u,v\rangle_S = \langle v,u\rangle_S$ *for all* $u,v \in S$.
3. $\langle \lambda u,v\rangle_S = \lambda\langle u,v\rangle_S$.

Theorem 2.1.1 (Bessel's inequality) *If $\{e_n\}$ is an orthonormal set in an inner product space S, then for any x in the space*

$$\sum_n |c_n|^2 \leqslant \|x\|^2$$

where $c_n = \langle x, e_n\rangle_S$.

Theorem 2.1.2 (Parseval's Relation) *Let $\{e_n\}$ be an orthonormal set in an inner product space S. It is a basis if and only if for each x in the space*

$$\sum_1^\infty |c_n|^2 = \|x\|^2,$$

where $\{c_n\}$ are the expansion coefficients of x with respect to $\{e_n\}$, $c_n = \langle x, e_n\rangle_S$.

Lemma 2.1.1 (Continuity of the Inner Product) *If $x_n \rightarrow x$, then $\langle x_n, y\rangle \rightarrow \langle x,y\rangle_S$ for any y. If $\sum_1^\infty u_n = U$, then $\sum_1^\infty \langle u_n, y\rangle_S = \langle U, y\rangle_S$ for any y.*

2.2 Calculus of Variation

2.2.1 Topologies on Banach Spaces

Let $(S, |\cdot|)$ denote a real Banach space. Let S' be the topological dual space of S defined as

$$S' = \Big\{ l : S \rightarrow \mathbb{R} \text{ is a linear function such that } |l|_{S'} = \sup_{x \neq 0} \frac{|l(x)|}{|x|_S} < \infty \Big\}.$$

Definition 2.2.1 (topologies on S) *(i) The strong topology, denoted by $x_n \underset{S}{\rightarrow} x$, is defined by*

$$|x_n - x|_S \rightarrow 0 (n \rightarrow +\infty).$$

(ii) The weak topology, denoted by $x_n \underset{S}{\rightharpoonup} x$, is defined as

$$l(x_n) \rightarrow l(x)(n \rightarrow +\infty) \quad \textit{for every } l \in S'.$$

(iii) The weak topology denoted by $l_n \overset{*}{\underset{S'}{\rightharpoonup}} l$, is defined by $l_n(x) \rightarrow l(x)$ $(n \rightarrow +\infty)$ for all $x \in S$.*

Remark 2.2.1 *Strong convergence implies weak convergence, but the converse is not true in general.*

Example 2.2.1 *Let $\{e_n\}$ be an infinite orthonormal set in an inner product space S. Then $\sum|\langle y,e_n\rangle|^2$ converges, and hence $\langle y,e_n\rangle \rightarrow 0$ for any $y \in S$. But $\|e_n\| = 1$, so $e_n \rightarrow 0$ is not true means no strong convergence.*

Theorem 2.2.1 *[7]*

1. *Let S be a reflexive[1] Banach space, $K > 0$ and $x_n \in S$ a sequence such that $|x_n|_S \leqslant K$. Then there exist $x \in S$ and a subsequence x_{n_j} of x_n such that $x_{n_j} \underset{S}{\rightharpoonup} x(n \rightarrow \infty)$.*

2. *Let S be a separable[2] Banach space, $K > 0$ and $l_n \in S'$ such that $|l_n|_{S'} \leqslant K$. Then there exist $l \in S'$ and a subsequence l_{n_j} of l_n such that $l_{n_j} \underset{S}{\overset{*}{\rightharpoonup}} l(n \rightarrow \infty)$.*

Let S be a Banach space, $F : S \rightarrow \mathbb{R}$ and consider the minimization problem

$$\min_{x\in S} F(x).$$

For existence of solution the following steps are to be achieved.

1. To construct a sequence $x_n \in S$ such that
$$\lim_{n\rightarrow\infty} F(x_n) = \min_{x\in S} F(x).$$

2. If F is coercive[3] then we can obtain a uniform bound $|x_n|_S \leqslant C$. If F is reflexive, then by theorem 2.2.1 we can deduce the existence of $x_0 \in S$ and of a subsequence x_{n_j} such that $x_{n_j} \underset{S}{\rightharpoonup} x_0$.

3. To prove that x_0 is a minimum point of F it suffices to have the inequality
$$\underline{\lim}_{x_{n_j} \underset{S}{\rightharpoonup} x_0} F(x_{n_j}) \geqslant F(x_0),$$
which implies that
$$F(x_0) = \min_{x\in S} F(x).$$

Definition 2.2.2 (Lower Semi Continuity) *F is called lower semi continuous (l.s.c) for the weak topology if for all sequence $x_n \rightharpoonup x_0$ we have*

$$\underline{\lim}_{x_n \rightharpoonup x_0} F(x_n) \geqslant F(x_0).$$

Definition 2.2.3 (Convexity) *F is said to be convex on S if*

$$F(\lambda x + (1-\lambda)y) \leqslant \lambda F(x) + (1-\lambda)F(y)$$

for all $x, y \in S$ and $\lambda \in [0,1]$. In other words we can say, a convex function is a continuous function whose value at the midpoint of every interval in its domain does not exceed the arithmetic mean of its values at the ends of the interval.

Example 2.2.2 (Examples on $\mathbb{R}$)

- *e^{ax}, for any $a \in \mathbb{R}$ on domain $\mathbb{R}$ is convex.*
- *x^α on $\mathbb{R}^+$, for $\alpha \geqslant 1$ or $\alpha \leqslant 0$ is convex.*

[1] The space S is said to be reflexive if $(S')'$ is isomorphic to S.

[2] A space S is separable if it contains a countable dense subset.

[3] F is said to be coercive if $\lim_{|x|\rightarrow+\infty} F = +\infty$.

- *Powers of absolute value $|x|^p$ on $\mathbb{R}$, for $p \geqslant 1$ are convex.*

Examples on $\mathbb{R}^n$:

- $f(x) = a^T x + b$ where $a \in \mathbb{R}^n, x, b \in \mathbb{R}^{n\times 1}$ is convex.
- Norms:

$$\|x\|_p = \Big(\sum_{i=1}^{n} |x_i|^p\Big)^{\frac{1}{p}},\ for\, p \geqslant 1; \qquad \|x\|_\infty = \max(|x_1|, |x_2|, \dots, |x_n|)$$

  are convex.

Examples on $\mathbb{R}^{m\times n}$:

- $$f(X) = tr\Big(A^T X\Big) + b = \sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}X_{ij} + b$$

  is convex.

- Spectral (maximum singular value) norm

$$f(X) = \|X\| = \sigma_{max}(X) = \Big(\lambda_{max}(X^T X)\Big)^{1/2}$$

  where λ_{max} is a maximum eigenvalue, is convex [13].

Theorem 2.2.2 *Let $F : S \longrightarrow \mathbb{R}$ be convex. Then F is weakly lower semi continuous if and only if F is strongly lower semi continuous.*

2.2.2 Gâteaux derivative of a functional

Assume that S and T are Banach spaces and $F : S \longrightarrow T$ be an Operator.

Definition 2.2.4 (Gâteaux derivative) *Let $u, v \in S$ then the quantity*

$$DF(u,v) = \lim_{h\to 0} \frac{F(u+hv) - F(u)}{h}$$

is called the G-differential of F at u in the direction of v. Moreover, if the above limit exists for any $v \in S$, the F is G-differentiable at point $u \in S$ and $DF(u,\cdot)$ is the G-derivative of F at u. If F is Gâteaux differentiable and if the problem $\min_{u\in S} F(u)$ has a solution u, then we have

$$DF(u,v) = 0.$$

Conversely, if F is convex, then a solution u of $DF(u,v) = 0\,\forall v \in S$ is a solution of the minimization problem. The equation $DF(u,v) = 0$ is called an Euler-Lagrange equation of the minimization problem $\min_{u\in S} F(u)$.

2.3 Total Variation (TV)and Bounded Variation (BV):

In this section we fist define TV, explain it with an example and then will give definition of BV. We use one-dimensional definition of BV for further explanation.

Definition 2.3.1 (Compact set) *A topological space S is compact if, for every collection $\{S_i\}_{i\in I}$ of open sets in S whose union is S, there exists a finite sub-collection $\{S_{i_j}\}_{j=1}^{n}$ whose union is also S.*

Definition 2.3.2 (Compact support) *Functions with compact support[4] in S are those with support that is a compact subset of S.*

Definition 2.3.3 *Let Ω be a bounded open subset of $\mathbb{R}^n$ and let u be a function in $L^1(\Omega)$, then defining the TV norm as*

$$\|Du\|(\Omega) = \int_\Omega |Du|(\mathrm{x})dx = \sup\left\{\int_\Omega u(\nabla\cdot v)dx : v = (v_1,\dots,v_n) \in C_0^\infty(\Omega), \quad |v|_{\ell_\infty} \leqslant 1\right\}, \tag{2.1}$$

where $\nabla \cdot v = \sum_{i=1}^{n} \frac{\partial v}{\partial x_i}$ and v is a vector valued function with compact support that is differentiable to arbitrary order. Here $C_0^\infty(\Omega)$ is the space of real-valued functions, infinitely continuous differentiable with compact support.

Definition 2.3.4 (Bounded Variation (BV)) *If for a function u, $\|Du\|(\Omega) < \infty$, then the function u is known as of bounded variation. The notation $BV(\Omega)$ denotes all functions in $L^1(\Omega)$ that are of bounded variation.*

To explain it more we give the one-dimensional definition of BV and give some examples

Definition 2.3.5 *A function $u : [a,b] \to \mathbb{R}$ is said to be of bounded variation on $[a,b]$ if and only if there exist a constant $M > 0$ such that*

$$\sum_{i=1}^{n} |u(x_i) - u(x_{i-1})| \leqslant M$$

for all partitions $P = \{x_0, x_1, \dots, x_n\}$ of $[a,b]$.

More details can be found in [133, 59].

Theorem 2.3.1 (Co-Area Formula) *Suppose that Ω is an open set in $\mathbb{R}^n$, let $u \in BV(\Omega)$ and define*

$$F_\eta = \{\mathrm{x} \in \Omega : u(\mathrm{x}) < \eta\},$$

be the level domain (cumulative level set). Then

$$\|Du\| = \int_\Omega |Du|dx = \int_{-\infty}^{\infty} d\eta \int_\Omega |D\chi_{F_\eta}|d\mathrm{x}, \tag{2.2}$$

where χ_{F_η} is a characteristic (or indicator) function of set F_η. The perimeter can be defined as

$$Per(F_\eta) = \int_\Omega |D\chi_{F_\eta}|dx,$$

so (2.2) becomes

$$\|Du\| = \int_\Omega |Du|dx = \int_{-\infty}^{\infty} Per(F_\eta)d\eta. \tag{2.3}$$

[4] The support of a function f is the closure of $\{x : f(x) \neq 0\}$ and is denoted by supp(f).

The proof could be found in [59].

Example 2.3.1 *If $u \in C^1(\Omega)$, then integration by parts gives*

$$\int_\Omega u\nabla \cdot v dx = -\int_\Omega \sum_{i=1}^{n} \frac{\partial u}{\partial x_i} v_i dx$$

for every $v \in C_0^\infty(\Omega, \mathbb{R}^n)$, so that

$$\int_\Omega |Du| d\mathbf{x} = \int_\Omega |\nabla u| dx,$$

where $\nabla u = \left(\frac{\partial u}{\partial x_1}, \frac{\partial u}{\partial x_2}, \ldots, \frac{\partial u}{\partial x_n}\right)$. Thus, if u is regular enough, the total variation (TV) has the form:

$$\|Du\|(\Omega) = \int_\Omega |Du|(\Omega) dx = \int_\Omega |\nabla u| dx \quad \mathit{for}\ u \in W^{1,1}(\Omega),$$

where $W^{1,p}(\Omega)$, for $1 \leqslant p < \infty$ is the Sobolev space of functions $u \in L^p(\Omega)$ such that all derivatives up to order 1 belong to $L^p(\Omega)$. $W^{1,\infty}(\Omega)$ is the space of locally Lipschitz functions.

Example 2.3.2 *Figure 2.1 illustrates a simple example [107], consider the characteristic function $u = \chi_L$ of a set L. Then*

$$\|Du\|(\Omega) = \|D\chi_L\|(\Omega) = Per(L, \Omega),$$

where $Per(L, \Omega)$ is the perimeter of L in Ω. For some η let us define the level sets for u:

$$L_\eta = \Omega \cap \{x : u(x) > \eta\}.$$

Then we have the following relation

$$\|Du\|(\Omega) = \int_{\mathbb{R}^1} Per(L_\eta, \Omega) d\eta.$$

This equation say that the total variation of a function u is computed by summing up the length of level lines over the contrast. For further illustration, let us consider a special case (Figure 2.2)

$$u = \begin{cases} u_1, & x \in \Omega_1 \\ u_2, & x \in \Omega_2. \end{cases} \tag{2.4}$$

Thus

$$\|Du\|(\Omega) = (u_2 - u_1)\partial(\Omega_1, \Omega_2),$$

where $\partial(\Omega_1, \Omega_2)$is the length of the boundary between Ω_1 and Ω_2 and also assume that $u_1 < u_2$. Thus the total variation measures contrast in images, and thus is a reasonable measure for signal transitions in real image data.

Example 2.3.3 *If $u : [a, b] \to \mathbb{R}$ is a monotonically increasing or decreasing, then for any partition $P = \{x_0, x_1, \ldots, x_n\}$ of $[a, b]$*

$$\sum_{i=1}^{n} |u(x_i) - u(x_{i-1})| = \sum_{i=1}^{n} [u(x_i) - u(x_{i-1})] = u(b) - u(a)$$

Thus u is of bounded variation on $[a, b]$.

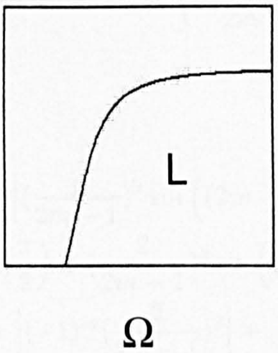


Figure 2.1: The total variation of the characteristic function χ_L measures the perimeter of the set L in Ω.

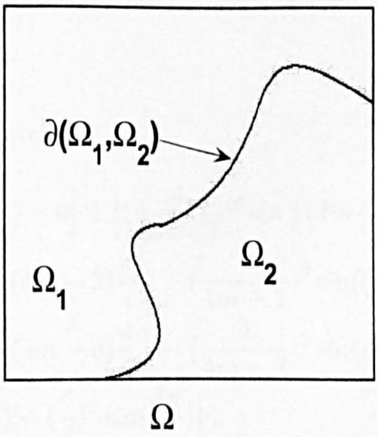


Figure 2.2: The total variation of a step function measures the length of the line of discontinuity times the contrast.

Example 2.3.4 *Define*

$$y = \begin{cases} 0 & \textit{if } x = 0 \\ x^2 \sin(\frac{\pi}{x}) & \textit{if } x \neq 0. \end{cases}$$

This function is of bounded variation on the interval $[0,2]$ *(see Fig:2.3). To see this mathematically let us take a partition* $P_m = \left\{0, \dfrac{2}{2m}, \dfrac{2}{2m-1}, \ldots, \dfrac{2}{3}, 1, 2\right\}$*, where* $m \in \mathbb{N}$*. Then*

$$\begin{aligned}
& \sum_{i=1}^{n} |y(x_i) - y(x_{i-1})| \\
= \; & \left|(\frac{2}{2m})^2 \sin(2m\frac{\pi}{2}) - 0\right| + \left|(\frac{2}{2m-1})^2 \sin\left((2m-1)\frac{\pi}{2}\right) - (\frac{2}{2m})^2 \sin(2m\frac{\pi}{2})\right| \\
+ \; & \left|(\frac{2}{2m-2})^2 \sin\left((2m-2)\frac{\pi}{2}\right) - (\frac{2}{2m-1})^2 \sin\left((2m-1)\frac{\pi}{2}\right)\right| + \ldots + \left|0 - (\frac{2}{3})^2 \sin(3\frac{\pi}{2})\right| + 4 \\
= \; & 0 + \left|(-1)^m (\frac{2}{2m-1})^2\right| + \left|(-1)^m (\frac{2}{2m-1})^2\right| + \ldots + \left|(\frac{2}{3})^2\right| + 4 \\
= \; & 2\left[\frac{4}{(2m-1)^2} + \frac{4}{(2m-3)^2} + \ldots + \frac{4}{3^2}\right] + 4 \\
< \; & 8\left[\frac{1}{(2m)^2} + \frac{1}{(2m-1)^2} + \frac{1}{(2m-2)^2} + \ldots + \frac{1}{3^2} + \frac{1}{2^2} + 1\right] + 4.
\end{aligned}$$

The series $\sum_{k=1}^{\infty} \frac{1}{k^2}$ *is convergent so for some* M*, we can write*

$$\sum_{i=1}^{n} |y(x_i) - y(x_{i-1})| < M.$$

Let us consider another partition

$$P_m = \left\{0, \frac{4}{4m}, \frac{4}{4m-1}, \frac{4}{4m-2}, \frac{4}{4m-3}, \frac{4}{4m-4}, \ldots, \frac{4}{5}, \frac{4}{4}, \frac{4}{3}, \frac{4}{2}\right\},$$

where $m \in \mathbb{N}$*. Then*

$$\begin{aligned}
& \sum_{i=1}^{n} |y(x_i) - y(x_{i-1})| \\
= \; & \left|(\frac{4}{4m})^2 \sin(4m\frac{\pi}{4}) - 0\right| + \left|(\frac{4}{4m-1})^2 \sin\left((4m-1)\frac{\pi}{4}\right) - (\frac{4}{4m})^2 \sin(4m\frac{\pi}{4})\right| \\
+ \; & \left|(\frac{4}{4m-2})^2 \sin\left((4m-2)\frac{\pi}{4}\right) - (\frac{4}{4m-1})^2 \sin((4m-1)\frac{\pi}{4})\right| \\
+ \; & \left|(\frac{4}{4m-3})^2 \sin\left((4m-3)\frac{\pi}{4}\right) - (\frac{4}{4m-2})^2 \sin((4m-2)\frac{\pi}{4})\right| \\
+ \; & \ldots + \left|2^2 \sin(\frac{2\pi}{4}) - (\frac{4}{3})^2 \sin(\frac{3\pi}{4})\right| \\
\leqslant \; & 16\sqrt{2}\left[\frac{1}{(4m-1)^2} + \frac{1}{(4m-3)^2} + \ldots + \frac{1}{3^2}\right] \\
+ \; & 16\left[\frac{1}{(4m-2)^2} + \frac{1}{(4m-6)^2} + \ldots \frac{1}{2^2}\right] \\
< \; & 16\sqrt{2}\left[\frac{1}{(4m)^2} + \frac{1}{(4m-1)^2} + \ldots + \frac{1}{2^2} + 1\right] \\
+ \; & 16\left[\frac{1}{(4m)^2} + \frac{1}{(4m-1)^2} + \ldots + \frac{1}{2^2} + 1\right].
\end{aligned}$$

Again the series $\sum_{k=1}^{\infty}\frac{1}{k^2}$ is convergent so for some M, we can write

$$\sum_{i=1}^{n}|y(x_i)-y(x_{i-1})|<M.$$

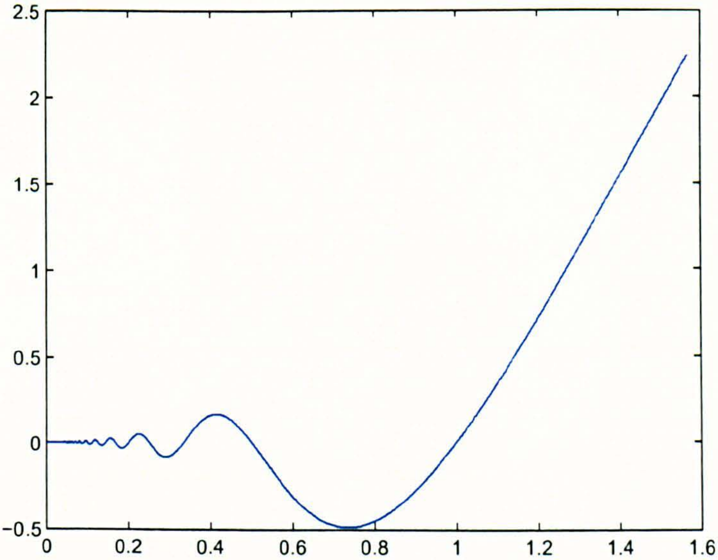


Figure 2.3: Function of Bounded Variation $y=x^2\sin(\frac{\pi}{x})$

Example 2.3.5 *Define the function*

$$y=\begin{cases}0 & \textit{if } x=0\\ x\cos(\frac{\pi}{x}) & \textit{if } x\neq 0.\end{cases}$$

This function is continuous, but is not of bounded variation (see Fig: 2.4) on the interval $[0,1]$*, because it oscillates more frequently near* $x=0$*. To see this let us take a partition* $P_m=\left\{0,\frac{1}{2m},\frac{1}{2m-1},\dots,\frac{1}{3},\frac{1}{2},1\right\}$*, where* $m\in\mathbb{N}$*. Then*

$$\begin{aligned}\sum_{i=1}^{n}|y(x_i)-y(x_{i-1})| &= |\frac{1}{2m}\cos(2m\pi)-0|+|\frac{1}{2m-1}\cos\Big((2m-1)\pi\Big)-\frac{1}{2m}\cos(2m\pi)|\\ &+|\frac{1}{2m-2}\cos\Big((2m-2)\pi\Big)-\frac{1}{2m-1}\cos\Big((2m-1)\pi\Big)|+\dots+|-1-\frac{1}{2}|\\ &=|\frac{1}{2m}|+|-\frac{1}{2m-1}-\frac{1}{2m}|+|\frac{1}{2m-2}+\frac{1}{2m-1}|+\dots+|-1-\frac{1}{2}|\\ &\geqslant 2[\frac{1}{2m}+\frac{1}{2m-1}+\dots+\frac{1}{2}]+1.\end{aligned}$$

The series $\sum_{k=2}^{\infty}\frac{1}{k}$ is divergent so for any M, there is a partition P_m for which

$$\sum_{i=1}^{n}|y(x_i)-y(x_{i-1})|>M.$$

More details and properties of BV space can be found in [7, 59, 133].

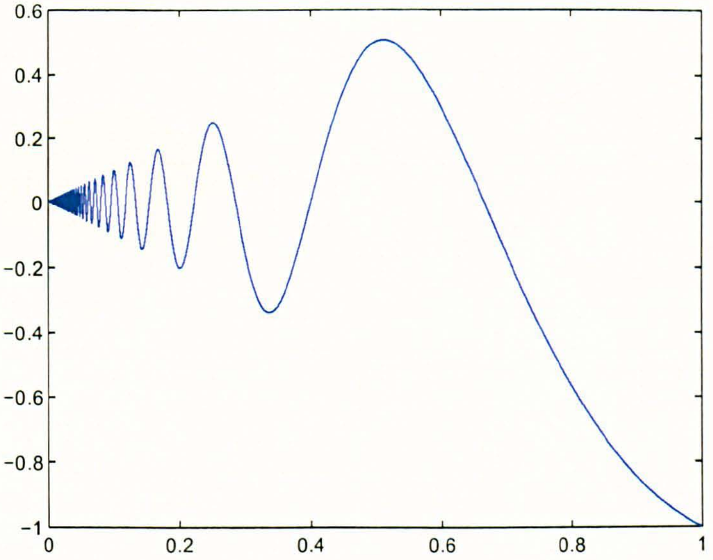


Figure 2.4: Function of Un-Bounded Variation $y = x\cos(\frac{\pi}{x})$

2.4 The Level Set Method

The level set method for tracking moving fronts was introduced by Osher and Sethian in [92]. This method has had a great success because it has been used in many applications in physics, from capturing multiphase fluid dynamics flows, to graphics, e.g. special effects in Movies, visualization, image processing, computer vision, control, visibility, ray tracing, segmentation, restoration and many others [83], [108]. In this section, we introduce the level set method applied to the theory of curve/surface evolution. We show that the level set formulation of a curve/surface evolution equation allows us to efficiently solve the problem of moving fronts, in particular the problem of changes of topology.

Let us consider the general geometric evolution of a curve Γ:

$$\begin{cases} \dfrac{\partial \Gamma}{\partial t} = V_{\perp}\mathcal{N} \\ \Gamma(t=0) = \Gamma_0 \end{cases} \tag{2.5}$$

where $V_{\perp}$ is the normal velocity and $\mathcal{N}$ is the unit normal to the curve Γ.

At this stage, we leave the parametric/explicit representation of a contour to interest in the geometry/implicit representation of the contour. This leads to the level set representation which is independent of the parametrization of the contour. The core idea in the level set method is to implicitly represent an interface Γ in $\mathbb{R}^2$ as a level set of a function ϕ, called level set function of higher dimension (in this case $\mathbb{R}^3$) and compute the geometric characteristics and the motion of the front with this level set function. The level set function ϕ of the closed front Γ is defined as follows [90]:

$$\begin{cases} \phi(x,y,t) > 0 \text{ inside } \Gamma \\ \phi(x,y,t) < 0 \text{ outside } \Gamma \\ \phi(x,y,t) = 0 \text{ on } \Gamma. \end{cases} \tag{2.6}$$

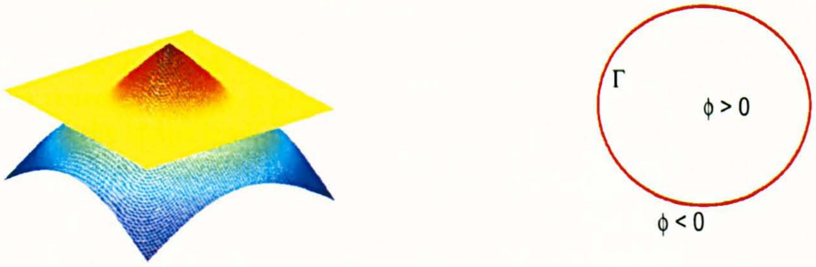


Figure 2.5: Plot of the higher dimension (3-D) function $\phi(x)$ and the interface $\Gamma = \{(x, y) \in \Omega : \phi(x, y) = 0\}$. Left: The level set function ϕ. Right: The interface Γ.

The geometric characteristics of the interface can be computed with the level set function. The unit normal $\mathcal{N}$ and the mean curvature κ to Γ are given by:

$$\begin{cases} \mathcal{N} = -\dfrac{\nabla\phi}{|\nabla\phi|} \\ \kappa = \nabla \cdot \mathcal{N} = \nabla \cdot \left(\dfrac{\nabla\phi}{|\nabla\phi|}\right). \end{cases} \tag{2.7}$$

The area of the region inside Γ and the length of the interface Γ are

$$\begin{cases} \int_{\mathbb{R}^2} H(\phi)dxdy \\ \int_{\mathbb{R}^2} |\nabla H(\phi)|dxdy = \int_{\mathbb{R}^2} \delta(\phi)|\nabla\phi|dxdy, \end{cases} \tag{2.8}$$

where δ and H are the Dirac delta and the Heaviside functions.

The motion of the front Γ evolving according to equation (2.5) is given by the evolution of the zero level set of ϕ which is solution of the following PDE:

$$\begin{cases} \dfrac{\partial\phi}{\partial t} = V_\perp|\nabla\phi| \\ \phi(t = 0) = d(\Gamma_0) = \phi_0, \end{cases} \tag{2.9}$$

where d is a function (usually a signed distance function) whose zero level set is the initial contour Γ_0.

Equation (2.9) raises a number of comments:

- The zero level set of Γ and all its level sets follows the front evolution equation (2.5).
- The level set evolution is computed on a fixed coordinate system since the level set is a parametrization free formulation.
- The evolution of the contour is independent of the initial embedding ϕ_0 [42, 53, 54, 55], and the classical solution, if it exists, of (2.5) coincides with the classical solution (2.9).

- Singularities can arise with PDE (2.9).

The level set method allows for curve topological changes, such as breaking or merging. Fig. 2.6 illustrates several important ideas about the level set method. In the upper-left corner we see a shape, that is, a bounded region with a well-behaved boundary. Below it, the red surface is the graph of a level set function ϕ determining this shape, and the flat blue region represents the x-y plane. The boundary of the shape is then the zero level set of ϕ, while the shape itself is the set of points in the plane for which ϕ is positive or zero.

In the top row we see a shape changing topology by splitting in two. It would be quite hard to describe this transformation numerically by parameterizing the boundary of the shape and following its evolution. One would need an algorithm able to detect the moment of the shape splitting in two, and then construct parameterizations for the two newly obtained curves. On the other hand, if we look at the bottom row, we see that the level set function merely got translated downward. Therefore we see that it is much easier to work with a shape through its level set function than with the shape directly, when we need to watch out for all the possible deformations the shape might undergo.

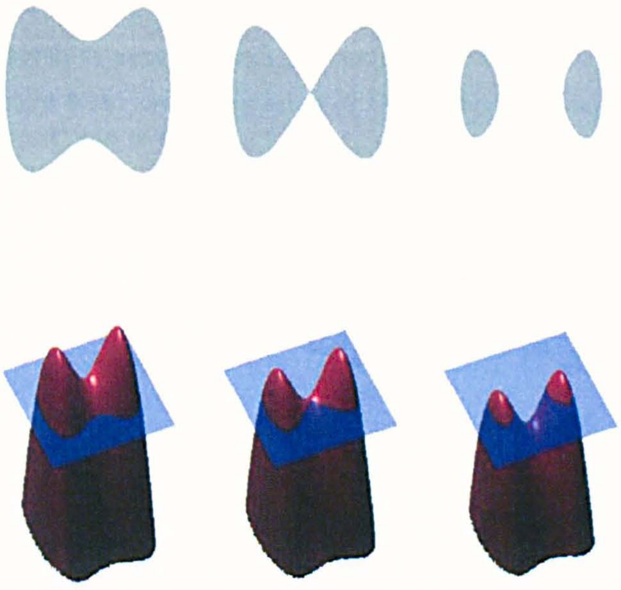


Figure 2.6: Illustration of how the level set function deal with topological changes. The top row presents the evolution of the curve and the bottom row shows the evolution of the associated level set function. We see that the curve changed its topology but not the level set function.

2.4.1 Numerical Implementation of The Level Set Method:

In this section we discuss the numerical implementation of PDE (2.9). In a wide range of important applications, $V_\perp$ has the form [90]:

$$V_\perp = \langle \vec{\Xi}(x,y), \frac{\nabla\phi}{|\nabla\phi|}\rangle - \xi(x,y)\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right), \tag{2.10}$$

where $\vec{\Xi}(x,y)$ is a vector field and $\xi(x,y)$ is a scalar field. Given the velocity $V_\perp$, (2.9) is a Hamilton-Jacobi equation whose solution can develop kinks (in absence of curvature term) which are discontinuities defined by jumps in derivatives. Special numerical methods are necessary to handle these discontinuities. These schemes presented in [92] based on upwind differencing, were then extended to higher order accuracy with the essentially non-oscillatory (ENO) and weighted essentially non-oscillatory (WENO) schemes in [91, 73]. Using equation (2.10), equation (2.9) becomes

$$\frac{\partial\phi}{\partial t} = \langle \vec{\Xi}(x,y), \nabla\phi\rangle - \xi(x,y)|\nabla\phi|\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right). \tag{2.11}$$

The first term of the right hand side of the above equation is a convection term. In the active contour framework, for example in geodesic active contour [21], the function $\vec{\Xi}(x,y)$ represents either an attraction force toward the boundaries of objects, i.e $\vec{\Xi}(x,y) = \nabla g$ where g is an edge detector function usually given by $g(\nabla(z*G_\sigma)) = \dfrac{1}{1+\gamma|\nabla(z*G_\sigma)|^2}$, (further details are given in next chapter) or a balloon force such as $\vec{\Xi} = -\Xi\dfrac{\nabla\phi}{|\nabla\phi|}$. The second term of the right hand side of equation (2.11) is a contour smoothing term based on the curvature of level sets of ϕ.

Let us consider the balloon term

$$\langle \vec{\Xi}, \nabla\phi\rangle = \langle -\Xi\frac{\nabla\phi}{|\nabla\phi|}, \nabla\phi\rangle = -\Xi|\nabla\phi|.$$

Discretization gives the following numerical scheme

$$-\Xi|\nabla\phi| = (-\Xi(x,y))^+D^+ + (-\Xi(x,y))^-D^-, \tag{2.12}$$

where

$$D^+ = \sqrt{((\Delta_x^-\phi)^+)^2 + ((\Delta_x^+\phi)^-)^2 + ((\Delta_y^-\phi)^+)^2 + ((\Delta_y^+\phi)^-)^2}$$
$$D^- = \sqrt{((\Delta_x^+\phi)^+)^2 + ((\Delta_x^-\phi)^-)^2 + ((\Delta_y^+\phi)^+)^2 + ((\Delta_y^-\phi)^-)^2},$$

$(\cdot)^+ = \max(\cdot, 0), (\cdot)^- = \min(\cdot, 0)$ and

$$\Delta_x^\pm\phi = \pm\frac{\phi(x\pm h_1, y) - \phi(x,y)}{h_1}, \Delta_y^\pm\phi = \pm\frac{\phi(x, y\pm h_2) - \phi(x,y)}{h_2},$$

h_1 and h_2 are horizontal and vertical spatial step sizes respectively. To illustrate this numerical scheme, let us consider $\Xi(x,y) = \pm 1$, in Figure 2.7 a circular curve is propagating inward and outward. The blue curve is the original curve and the red curve is the propagated curve after applying the numerical scheme (2.12) on balloon term.

The second term of the right hand side of equation (2.11) is a regularization term based on the mean curvature. This term is parabolic and therefore it does not need an

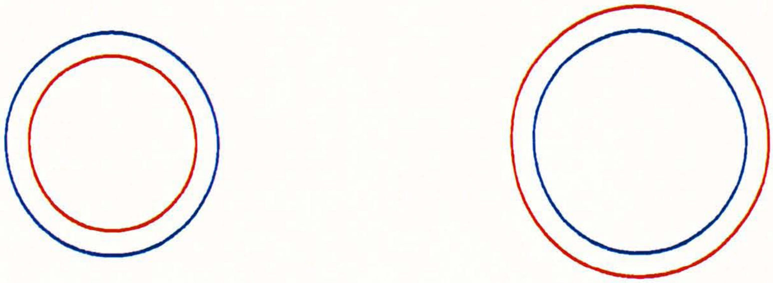


Figure 2.7: Evolution of a circular contour inward (left) and outward (right). Left: $\Xi(x, y) = -1$ propagating inward, blue contour is the initial and the red contour is final propagated contour. Right: $\Xi(x, y) = 1$, propagating outward, blue contour is the initial and red is the final propagated contour. These results are obtained by implementation of equation (2.12).

upwind scheme which has been designed for hyperbolic advection term to guide the propagation directions. For a parabolic term, the propagation is in all direction, hence the central difference approximation scheme fits well to approximate the term $\xi|\nabla\phi|\nabla \cdot \left(\frac{\nabla\phi}{|\nabla\phi|}\right)$ at a first order of accuracy. In 2-D images, the curvature of the level sets is

$$\nabla \cdot \left(\frac{\nabla\phi}{|\nabla\phi|}\right) = \frac{\phi_{xx}\phi_y^2 - 2\phi_x\phi_y\phi_{xy} + \phi_{yy}\phi_x^2}{(\phi_x^2 + \phi_y^2)^{3/2}} \tag{2.13}$$

and the associated numerical scheme is

$$\xi|\nabla\phi|\nabla \cdot \left(\frac{\nabla\phi}{|\nabla\phi|}\right) = \xi\frac{\Delta_{xx}^0\phi(\Delta_y^0\phi)^2 - 2\Delta_x^0\phi\Delta_y^0\phi\Delta_{xy}^0\phi + \Delta_{yy}^0\phi(\Delta_x^0\phi)^2}{((\Delta_x^0)^2 + (\Delta_y^0)^2)^{1/2}}, \tag{2.14}$$

where

$$\begin{aligned}
\Delta_x^0\phi &= (\phi(x+h_1, y) - \phi(x-h_1, y))/2h_1,\\
\Delta_{xx}^0\phi &= (\phi(x+h_1, y) - 2\phi(x, y) + \phi(x-h_1, y))/h_1^2,\\
\Delta_y^0\phi &= (\phi(x, y+h_2) - \phi(x, y-h_2))/2h_2,\\
\Delta_{yy}^0\phi &= (\phi(x, y+h_2) - \phi(x, y) + \phi(x, y-h_2))/h_2^2\\
\text{and}&\\
\Delta_{xy}^0\phi &= (\phi(x+h_1, y+h_2) + \phi(x-h_1, y-h_2) - \phi(x+h_1, y-h_2) - \phi(x-h_1, y+h_2))/4h_1h_2.
\end{aligned}$$

This smoothing term is tested on a non smooth curve in figure 2.8 where the left part is the initial contour of the level set function ϕ_0 and the right part is the smooth contour, when regularization term is applied.

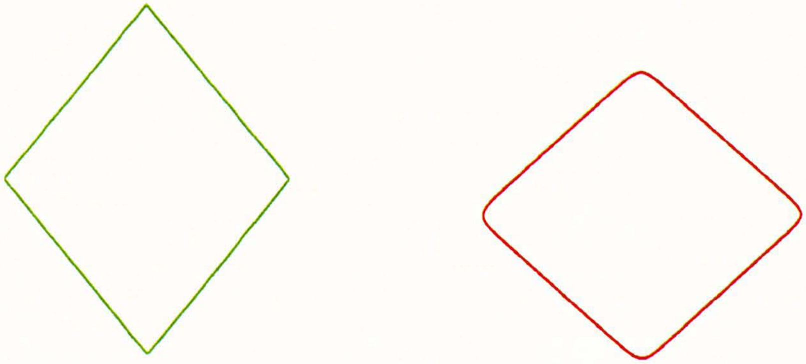


Figure 2.8: Smoothing of contours represented by the zero level set of a level set function. Left: Initial contour ϕ_0. Right: Smooth contour after applying the curvature dependent regularizer.

2.4.2 Re-initialization

If the level set function ϕ becomes too steep or too flat, the error increases when calculating the derivatives numerically. Further a small change in ϕ will lead to a large change of the zero level set if the level set function is too flat around zero. Therefore, it can be useful for the numerical calculations to re-initialize the level set function. The idea is to change ϕ into the distance function such that $|\nabla\phi| = 1$ for all x , y. The re-initialization changes the level set function everywhere, except from the zero level set which must be the same before and after re-initialization. This explains why the re-initialization does not change the mathematical problem, since the regions of interest are where $\phi > 0$ and where $\phi < 0$, and these areas do not change through the re-initialization. Solve the following PDE for $\phi(x, y, t)$

$$\begin{aligned} &\phi_t - \operatorname{sgn}(\phi)(1 - |\nabla\phi|) = 0 \qquad \text{in } \mathbb{R} \times (0, \infty) \\ &\phi(x, y, 0) = \phi_0 \qquad \text{in } \mathbb{R} \end{aligned} \tag{2.15}$$

where ϕ_0 is the function which is supposed to be re-initialized and $\operatorname{sgn}(\cdot)$ is a signum function. These equations will convert the level set function to the unit distance function.

2.5 Inverse and Ill-Posed Problems and Regularization

Inverse problems arise in many areas of scientific computing. Very often inverse problems are formulated in such way that inner properties of a system can be deduced from exterior measurements by solving the inverse problem. The word inverse is used to indicate that the problem has a corresponding forward or direct problem. If the direct problem is formulated as

$$\mathcal{K} : S_1 \to S_2$$

for some appropriate spaces S_1 and S_2, then the inverse problem is formulated as

$$\mathcal{K}^{-1} : S_2 \to S_1.$$

To illustrate this concept, consider the Fredholm integral equation. This integral equation arise in numerous applications, including image processing. The one-dimensional Fredholm integral equation (of the first kind) is

$$\int_a^b k(s,t)u(t)dt = g(s), \qquad 0 \leqslant s \leqslant 1. \tag{2.16}$$

$k(s,t)$ is called the kernel of the integral equation. The forward problem is formulated as follows:
Given $k(s,t)$ and $u(t)$, find $g(s)$. The solution of this problem is done in the obvious way, by evaluation of the integral. There are three different inverse problems corresponding to the forward problem, depending on the information already known.

1. Given $g(s)$, find $k(s,t)$ and $u(t)$,
2. Given $g(s)$ and $k(s,t)$, find $u(t)$, or
3. Given $g(s)$ and $u(t)$, find $k(s,t)$.

If the kernel is in the special form $k(s,t) = k(s-t)$, the three problems above are called de-convolution problems. The most challenging problem is to recover both $k(s,t)$ and $u(t)$ from measurements of $g(s)$ which is known as blind de-convolution.

2.5.1 Ill-Posed Problems

Definition 2.5.1 (Well-Posed problem) *Let $\mathcal{K} : S_1 \to S_2$, where S_1 and S_2 are Hilbert spaces*[5] *then the problem*

$$\mathcal{K}(u) = b \tag{2.17}$$

is said to be well-posed, in the sense of Hadamard, if

1. *For each $b \in S_2$ there exist a solution $u \in S_1$ such that $\mathcal{K}(u) = b$ holds.*
2. *The solution is unique.*
3. *The solution is stable with respect to perturbations in the data, that is given $u^* \in S_1$ and $b^* \in S_2$ such that $\mathcal{K}(u^*) = b^*$ then for every $\epsilon > 0$ there exists a $\delta_1 > 0$ such that $\|\mathcal{K}(u) - b^*\| < \delta_1$ implies that $\|u - u^*\| < \epsilon$.*

A problem which is not well-posed is called ill-posed.

The first condition tells that $\mathcal{K}$ is onto, i.e. the range of $\mathcal{K}$, $R(\mathcal{K})$ is equal to S_2. In the case of linear operator, the second condition is equivalent to requiring that $\mathcal{N}(\mathcal{K}) = \{0\}$ where $\mathcal{N}(\mathcal{K})$ is the null space of $\mathcal{K}$. If the first two conditions hold then the inverse of $\mathcal{K}$ exists and the third condition says that the inverse should be continuous. For the case when $\mathcal{K}$ is linear, well posedness is equivalent to the requirement that the inverse operator $\mathcal{K}^{-1} : S_2 \to S_1$, exists and is bounded.

Definition 2.5.2 (Orthogonal Complement) *If $E \subset S_i$ then $u \in S_i$ is an element of the orthogonal complement of E if and only if $\langle u, v\rangle_i = 0$ for all $v \in E$. The orthogonal complement will be denoted by $E^\perp$.*

[5] A complete space with a positive definite inner product is known as a Hilbert space

Definition 2.5.3 (Adjoint Operator) *Let $\mathcal{K}$ be a linear operator with a dense domain in S_1 mapping into S_2. The adjoint operator $\mathcal{K}^* : S_2 \to S_1$ is a linear operator where for every $y \in \mathcal{D}(\mathcal{K}^*)$, $\mathcal{D}(\mathcal{K}^*)$ is the domain of $\mathcal{K}^*$, there exists a unique $y^* \in S_1$ such that*

$$\langle \mathcal{K}u, y\rangle_2 = \langle u, y^*\rangle_1,$$

for every $u \in \mathcal{D}(\mathcal{K})$. The adjoint is defined by the mapping $\mathcal{K}^*y = y^*$ for all $y \in \mathcal{D}(\mathcal{K}^*)$.

When $\mathcal{K}$ is not 1-1 the pseudo-inverse can be used to resolve the non-uniqueness [51].

Definition 2.5.4 (Pseudo-inverse) *Let $\mathcal{K}$ be a continuous linear operator. Define $\tilde{\mathcal{K}}$ by $\tilde{\mathcal{K}} = \mathcal{K} \mid_{\mathcal{N}(\mathcal{K})^\perp}$, thus $\tilde{\mathcal{K}}$ is 1-1 and maps onto the range of $\mathcal{K}$. The pseudo-inverse, $\mathcal{K}^\dagger$, is the linear extension of $\tilde{\mathcal{K}}^{-1}$ to the domain*

$$\mathcal{D}(\mathcal{K}^\dagger) = R(\mathcal{K}) + R(\mathcal{K})^\perp,$$

with minimal operator norm.

It is worth mentioning that for any $b_0 \in \mathcal{D}(\mathcal{K}^\dagger)$ the pseudo-inverse gives the best least squares approximation to b_0 [51], i.e. $u_0 = \mathcal{K}^\dagger b_0$ satisfies

$$\|u_0\|_1 = \inf_{u \in \mathcal{D}(\mathcal{K})} \left(\|u\|_1 \middle| \|\mathcal{K}u - b_0\|_2 = \inf_{v \in \mathcal{D}(\mathcal{K})} \|\mathcal{K}v - b_0\| \right).$$

The problem in (2.17) is an ill-posed problem when $\mathcal{K}^\dagger$ is unbounded. One type of operator that can have an unbounded pseudo-inverse is a compact operator.

Definition 2.5.5 (Compact Operator) *An operator $\mathcal{K}$ is compact if and only if the image of any bounded set is a relatively compact set.*

Definition 2.5.6 *A relatively compact set is a set whose closure is compact.*

An example of a compact operator is the Fredholm integral defined in equation (2.16).

Theorem 2.5.1 *For a compact operator whose range is infinite dimensional the pseudo-inverse operator is densely defined and unbounded.*

Proof of this theorem can be found in [51]. With an unbounded pseudo-inverse the problem in (2.17) is ill-posed since it violates first and third properties of the well-posdness. A special tool that allows for further analysis of a compact linear operator is the singular value expansion [51].

Theorem 2.5.2 *Let $\mathcal{K} : S_1 \to S_2$ be a compact linear operator. Then there exists positive values σ_n, with associated functions $\psi_n \in S_1$ and $\varphi_n \in S_2$ such that $\mathcal{K}\psi_n = \sigma_n\varphi_n$ and $\mathcal{K}^*\varphi_n = \sigma_n\psi_n$. If $\mathcal{K}$ has infinite dimensional range then $n = 1, 2, \ldots$, otherwise there will be finitely many terms. The functions $\{\psi_n\}$ are an orthonormal basis of $\mathcal{N}(\mathcal{K})^\perp$, and the functions $\{\varphi_n\}$ are an orthonormal basis of the closure of $R(\mathcal{K})$. Furthermore,*

$$\mathcal{K}u = \sum_n \sigma_n \langle u, \psi_n\rangle_1 \varphi_n \tag{2.18}$$

and

$$\mathcal{K}^*b = \sum_n \sigma_n \langle b, \varphi_n\rangle_2 \psi_n, \tag{2.19}$$

for all $u \in S_1$, and $b \in S_2$. The expression in (2.18) is known as the singular value expansion of $\mathcal{K}$.

The σ_n's are referred to as the *singular values* of $\mathcal{K}$. For a compact operator the singular values can be arranged in descending order, $\sigma_1 \geqslant \sigma \geqslant \ldots > 0$. The collection $\{\sigma_n; \varphi_n, \psi_n\}$ is known to be the *singular system* for $\mathcal{K}$.

Theorem 2.5.3 *Let $\mathcal{K}$ have a singular system given by $\{\sigma_n; \varphi_n, \psi_n\}$. Then the pseudo-inverse has the following expansion*

$$\mathcal{K}^{\dagger} b = \sum_n \frac{\langle b, \varphi_n \rangle_2}{\sigma_n} \psi_n, \tag{2.20}$$

for all $b \in \mathcal{D}(\mathcal{K}^{\dagger})$.

This theorem gives the singular value expansion for the pseudo-inverse.

Theorem 2.5.4 *Let $\mathcal{K}$ satisfy the hypothesis of Theorem 2.5.2 with the additional property that the range is infinite dimensional. Then*

$$\lim_{n \to \infty} \sigma_n = 0.$$

The expansion in (2.19) reveals the lack of continuous dependence of the pseudo-inverse solution when the range of $\mathcal{K}$ is infinite dimensional. If there is any error present in φ_n, ψ_n or b, then that error may be greatly amplified by the division of the small singular values of $\mathcal{K}$.

2.5.2 Regularization

In several fields of mathematics, in particular statistics, machine learning and inverse problems, regularization involves introducing additional information in order to solve an ill-posed problem or prevent over-fitting. This information is usually of the form of a penalty for complexity, such as restrictions for smoothness or bounds on the vector space norm.

A theoretical justification for regularization is that it attempts to impose Occam's razor[6] on the solution. From a Bayesian point of view, many regularization techniques correspond to imposing certain prior distributions on model parameters.

The same idea arose in many fields of science. For example, the least-squares method can be viewed as a very simple form of regularization. A simple form of regularization applied to integral equations, generally termed Tikhonov regularization

Definition 2.5.7 (Regularization Operator) *A regularization operator for $\mathcal{K}$ is a one parameter family of continuous operators $R_\alpha : S_2 \longrightarrow S_1$ such that for $\mathcal{K}u_0 = b_0$ the following conditions hold:*

1. *There exist numbers α_0 and δ_0 such that $R_\alpha(b)$ is defined for all $0 < \alpha < \alpha_0$ and $\|b - b_0\| < \delta_0$.*

2. *There exists a function $\alpha(\delta)$ such that given any $\epsilon > 0$ there is a number $\delta(\epsilon) < \delta_0$ such that if $\|b - b_0\| < \delta(\epsilon)$ then $\|u_\gamma - u_0\|_1 < \epsilon$ where $u_\gamma = R_\gamma(b)$ and $\gamma = \alpha(\delta(\epsilon))$.*

Notice that if $\mathcal{K}$ is linear then the first condition is equivalent to the requirement that R_α is bounded and the second condition is equivalent to the requirement that $R_\alpha(b) \longrightarrow \mathcal{K}^{\dagger} b$ as $\alpha \longrightarrow 0$ for all $b \in \mathcal{D}(\mathcal{K}^{\dagger})$ [51]. There are many different regularization methods that can be applied to an ill-posed problem. For more details about regularization methods see [51, 64, 117].

[6] Occam's razor principle states that the explanation of any phenomenon should make as few assumptions as possible, eliminating those that make no difference in the observable predictions of the explanatory hypothesis or theory.

2.5.3 Tikhonov Regularization

In an abstract Hilbert space setting (i.e. $\mathcal{K} : S_1 \to S_2$), Tikhonov regularization is given by

$$u_\alpha = R_\alpha(b) = \text{argmin}_{u \in \mathcal{D}(\mathcal{K})} \left(\frac{1}{2} \|\mathcal{K}u - b\|_2^2 + \frac{\alpha}{2} \|u\|_1^2 \right), \tag{2.21}$$

where $\text{argmin}_{u \in \mathcal{D}(\mathcal{K})}$ denotes an element out of $\mathcal{D}(\mathcal{K})$ that obtains the minimum value for the given functional. This method can be thought of as penalized least squares with the second term in (2.21) being the penalty term. The regularization parameter, $\alpha > 0$, is used to assign a weight to the penalty term. Finding the u_α that minimizes the functional in (2.21) involves the use of calculus of variation. Consequently, we will need the first variation of the functional $F(u) = \frac{1}{2}\|\mathcal{K}u - b\|_2^2 + \frac{\alpha}{2}\|u\|_1^2$ in the direction of $v \in S_1$. The first variation is given by

$$\begin{aligned} F'(u; v) &\equiv \lim_{h \to 0} \left(\frac{F(u + hv) - F(u)}{h} \right) \\ &= \lim_{h \to 0} \frac{1}{2h} (\langle \mathcal{K}(u + hv) - b, \mathcal{K}(u + hv) - b \rangle_2 + \alpha \langle u + hv, u + hv \rangle_1) \\ &= \langle \mathcal{K}u - b, \mathcal{K}v \rangle_2 + \alpha \langle u, v \rangle_1. \end{aligned} \tag{2.22}$$

Using the adjoint in (2.22) and the fact that at the minimum the first variation must be zero for all $v \in S_1$ gives a representation for the solution to (2.21),

$$u_\alpha = (\mathcal{K}^*\mathcal{K} + \alpha I)^{-1} \mathcal{K}^* b. \tag{2.23}$$

If $\mathcal{K}$ has the singular system $\{\sigma_n; \varphi_n, \psi_n\}_{n=1}^\infty$ then the solution in (2.23) has a singular value expansion

$$u_\alpha = \sum_{n=1}^\infty \frac{\sigma_n}{\sigma_n^2 + \alpha} \langle b, \varphi_n \rangle_2 \psi_n. \tag{2.24}$$

Using the condition $\alpha > 0$, the expansion in (2.24), and the fact that both the φ_n's and ψ_n's are an orthonormal basis results in

$$\begin{aligned} \|R_\alpha\|_1 &= \sqrt{\sum_{n=1}^\infty \left| \frac{\sigma_n}{\sigma_n^2 + \alpha} \langle b, \varphi_n \rangle_2 \right|^2} \\ &\leqslant \frac{\sigma_1}{\alpha} \sqrt{\sum_{n=1}^\infty |\langle b, \varphi_n \rangle_2|^2} \\ &\leqslant \frac{\sigma_1}{\alpha} \|b\|_2. \end{aligned}$$

Hence R_α is a bounded operator for each $\alpha > 0$. For any fixed n it is also true that $\lim_{\alpha \to 0} \frac{\sigma_n}{\sigma_n^2 + \alpha} = \frac{1}{\sigma_n}$, which matches the expansion in (2.20). Therefore, (2.21) defines a regularization operator.

In this thesis we discuss inverse problems in image processing. If the regularization functional penalizes non-smooth images, the effect of this regularization will be noise removal but also a smoothing of the edges in the image. To overcome this disadvantage with classical noise removal techniques L. I. Rudin, et al (ROF) in their seminal 1992 paper [99] introduced the Total Variation (TV) regularization functional

$$|u|_{TV} = \int_\Omega |\nabla u| dxdy \tag{2.25}$$

where Ω is the image domain as discussed in Section 2.3. The TV regularization functional does not distinguish between smooth and piecewise smooth solutions with the same total variation, and thus Tikhonov regularization with the TV regularization functional can remove noise while still preserving the edges in an image. Note that the Euclidean norm $|\cdot|$ is not differentiable at the origin. To overcome the resulting numerical difficulties, (2.25) is replaced by the following

$$|u|_{TV_\beta} = \int_\Omega \sqrt{|\nabla u|^2 + \beta^2} dxdy, \tag{2.26}$$

where $\beta > 0$ is a small perturbing parameter. Thus Tikhonov regularization with the perturbed TV regularization functional involves the solution of the minimization problem

$$\min_{u \in BV(\Omega)} \frac{1}{2} \int_\Omega (\mathcal{K}u - b)^2 dxdy + \alpha \int_\Omega \sqrt{|\nabla u|^2 + \beta^2} dxdy. \tag{2.27}$$

First variation leads to the following Euler-Lagrange equation

$$-\alpha \nabla . \left(\frac{\nabla u}{\sqrt{|\nabla u|^2 + \beta}} \right) + \mathcal{K}^* \mathcal{K} u = \mathcal{K} b. \tag{2.28}$$

This equation is highly nonlinear and the efficient solution of the discrete version of this equation using iterative methods in both the deblurring and the pure denoising cases has been an active area of research over the last decade. In this thesis we mainly discussed methods used for solving this type of and other related problems. More details about TV regularization can be found in [112, 113] and solution of TV and BV based image processing problems can be found in [69, 71, 70, 99].

2.6 Iterative Methods for Solving System of Equations

In this section we present some basic iterative methods for solving a linear system of equations $\mathbf{Ax} = \mathbf{b}$ where $\mathbf{A}$ is a given $m_1 \times m_1$ matrix, $\mathbf{b}$ is a given vector of size m_1. An iterative technique to solve the linear system $\mathbf{Ax} = \mathbf{b}$ starts with an initial approximation $\mathbf{x}^{(0)}$ to the solution $\mathbf{x}$ and generates a sequence of vectors $\{\mathbf{x}^{(k)}\}_{k=0}^{\infty}$ that converges to $\mathbf{x}$. All iterative techniques convert the system $\mathbf{Ax} = \mathbf{b}$ into an equivalent system of the form $\mathbf{x} = \mathbf{Tx} + \mathbf{c}$ for some fixed matrix $\mathbf{T}$ and a vector $\mathbf{c}$. After the initial vector $\mathbf{x}^{(0)}$ is selected, the sequence of approximate solution vectors is generated by computing

$$\mathbf{x}^{(k)} = \mathbf{T}\mathbf{x}^{(k-1)} + \mathbf{c} \tag{2.29}$$

for each $k = 1, 2, 3, \ldots$.

2.6.1 The Jacobi Iterative Method

Consider a system of equations

$$\mathbf{Ax} = \mathbf{b} \tag{2.30}$$

where $\mathbf{A} = [a_{ij}]_{m_1 \times m_1}$ and $\mathbf{b} = [b_i]_{m_1 \times 1}$. Solving ith equation for x_i we have the equivalent equations

$$x_i = \sum_{\substack{j=1 \\ j \neq i}}^{m_1} \left(\frac{-a_{ij} x_j}{a_{ii}} \right) + \frac{b_i}{a_{ii}}$$

generating each $x_i^{(k)}$ from components of $\mathbf{x}^{(k-1)}$ for $k \geqslant 1$ by

$$x_i^{(k)} = \frac{1}{a_{ii}} \Big(\sum_{\substack{j=1 \\ j \neq i}}^{m_1} (-a_{ij} x_j^{(k-1)}) + b_j \Big), \qquad \text{for } i = 1, 2, \ldots, m_1. \tag{2.31}$$

To write (2.30) in the form (2.29), express the matrix $\mathbf{A}$ in the form

$$\mathbf{A} = \mathbf{D} - \mathbf{L} - \mathbf{U}$$

where $\mathbf{D}, -\mathbf{L}, -\mathbf{U}$ are diagonal, strictly lower triangular and strictly upper triangular matrices respectively of matrix $\mathbf{A}$. We have that

$$\begin{aligned} &(\mathbf{D} - \mathbf{L} - \mathbf{U})\mathbf{x} = \mathbf{b} \\ \iff\ & \mathbf{D}\mathbf{x} = (\mathbf{L} + \mathbf{U})\mathbf{x} + \mathbf{b} \\ \iff\ & \mathbf{x} = \mathbf{D}^{-1}(\mathbf{L} + \mathbf{U})\mathbf{x} + \mathbf{D}^{-1}\mathbf{b} \end{aligned}$$

thus in (2.29) form, the Jacobi method is

$$\mathbf{x} = \mathbf{T_J}\mathbf{x} + \mathbf{c_J}$$

where

$$\mathbf{T_J} = \mathbf{D}^{-1}(\mathbf{L} + \mathbf{U})$$

and

$$\mathbf{c_J} = \mathbf{D}^{-1}\mathbf{b}.$$

Algorithm for Jacobi Method: To solve (2.30) with initial approximation $\mathbf{x}^{(0)}$: INPUT the number of equations and unknowns m_1; the entries a_{ij} of the matrix $\mathbf{A}$; the entries b_j of b; the entries of $\mathbf{x}^{(0)}$ and the tolerance TOL; maximum number of iterations N and also let $\mathbf{X0}$=$\mathbf{x}^{(0)}$.

```
Algorithm 1 (Jacobi Method) Step 1   Set $k = 1$.
Step 2   While ($k \leqslant N$) do steps 3-6.
Step 3   For $i = 1, 2, \ldots, m_1$ set
             $x_i = \sum_{j=1, j \neq i}^{m_1} \left( \frac{-a_{ij}\mathbf{X0}_j}{a_{ii}} \right) + \frac{b_i}{a_{ii}}$
Step 4   If $\|\mathbf{x} - \mathbf{X0}\| < TOL$ then OUTPUT $(x_1, \ldots, x_{m_1})$;
                          end.
Step 5   Set $k = k + 1$.
Step 6   For $i = 1, \ldots, m_1$ set $\mathbf{X0}_i = x_i$.
Step 7   OUTPUT ('Maximum number of iterations exceeded');
```

end.

Step 3 of the algorithm requires that $a_{ii} \neq 0$ for each $i = 1, 2, \ldots, m_1$. If one of the a_{ii} entries is zero and the matrix is nonsingular, a reordering of the equations can be performed so that $a_{ii} \neq 0$. To speed up the convergence, the equations should be arranged so that a_{ii} is as large as possible.

Weighted Jacobi Method: In the Weighted Jacobi Method, given the current approximation $x^{(k-1)}$ the new Jacobi iterates are computed using

$$x_i^* = \frac{1}{a_{ii}}\Big(\sum_{\substack{j=1\\ j\neq i}}^{m_1}(-a_{ij}x_j^{(k-1)}) + b_j\Big), \qquad for\, i = 1, 2, \ldots, m_1. \tag{2.32}$$

for $i = 1, 2, \ldots, m_1$ as before, however x^* is now just an intermediate value. The new approximation $x^{(k)}$ is given by:

$$x^{(k)} = (1-\omega)x_i^{(k-1)} + \omega x^* \tag{2.33}$$

where ω is a weighting factor to be chosen. Of course when $\omega = 1$ we have the original Jacobi Method. In matrix form the weighted Jacobi Method is:

$$x^{(k)} = ((1-\omega)\mathbf{I} + \omega\mathbf{T_J})x^{(x-1)} + \omega\mathbf{D}^{-1}\mathbf{b} \tag{2.34}$$

which is equivalent to

$$\mathbf{x^{(k)}} = \mathbf{T}_\omega\mathbf{x^{(k-1)}} + \mathbf{c}_\omega$$

where $\mathbf{T}_\omega = (1-\omega)\mathbf{I} + \omega\mathbf{D}^{-1}(\mathbf{L}+\mathbf{U})$ and $\mathbf{c}_\omega = \omega\mathbf{D}^{-1}\mathbf{b}$.

2.6.2 Gauss Seidel Method

In Jacobi method, when we compute $x_i^{(k)}$ we have already computed $x_1^{(k)}, \ldots, x_{i-1}^{(k)}$ which should be better approximations to $x_1, \ldots, x_{i-1}$ than $x_1^{(k-1)}, \ldots, x_{i-1}^{(k-1)}$. Thus we have

$$x_i^{(k)} = \frac{1}{a_{ii}}\Big(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^{m_1} a_{ij}x_j^{(k-1)}\Big) \quad \text{for } i = 1, \ldots, m_1, \tag{2.35}$$

which is known as the Gauss-Seidel Method.

To write the Gauss-Seidel method in matrix form, multiply equation (2.35) by a_{ii} we have

$$a_{ii}x_i^{(k)} + \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} = b_i - \sum_{j=i+1}^{m_1} a_{ij}x_j^{(k-1)}$$

or

$$a_{ii}x_i^{(k)} + \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} = b_i + \sum_{j=i+1}^{m_1}\Big(-a_{ij}x_j^{(k-1)}\Big)$$

thus in matrix notation we have

$$(\mathbf{D}-\mathbf{L})\mathbf{x}^{(k)} = \mathbf{U}\mathbf{x}^{(k-1)} + \mathbf{b}$$

or equivalently

$$\mathbf{x}^{(k)} = \mathrm{T}_{GS}\mathbf{x}^{(k-1)} + \mathbf{c}_{GS} \tag{2.36}$$

where

$$\mathbf{T}_{GS} = (\mathbf{D} - \mathbf{L})^{-1}\mathbf{U}$$

and

$$\mathbf{c}_{GS} = (\mathbf{D} - \mathbf{L})^{-1}\mathbf{b}.$$

Algorithm for Gauss-Seidel Method: To solve (2.30) with initial approximation $\mathbf{x}^{(0)}$: INPUT the number of equations and unknowns m_1; the entries a_{ij} of the matrix $\mathbf{A}$; the entries b_j of $\mathbf{b}$ the entries of $\mathbf{x}^{(0)}$ and the tolerance TOL; maximum number of iterations N and also let $\mathbf{X0}=\mathbf{x}^{(0)}$.

```
Algorithm 2 (Gauss-Seidel Method) Step 1   Set $k = 1$.
Step 2   While ($k \leqslant N$) do steps 3-6.
Step 3   For $i = 1, 2, \dots, m_1$ set
             $x_i = \frac{1}{a_{ii}}\Big(b_i - \sum_{j=1}^{i-1} a_{ij}x_j - \sum_{j=i+1}^{m_1} a_{ij}\mathbf{X0}_j\Big)$
Step 4   If $\|\mathbf{x} - \mathbf{X0}\| < TOL$ then OUTPUT $(x_1, \dots, x_{m_1})$;
                                end.
Step 5   Set $k = k + 1$.
Step 6   For $i = 1, \dots, m_1$ set $\mathbf{X0}_i = x_i$.
Step 7   OUTPUT ('Maximum number of iterations exceeded');
                                end.
```

As with Jacobi method step 3 of the algorithm requires that $a_{ii} \neq 0$ for each $i = 1, 2, \dots, m_1$. If one of the a_{ii} entries is zero and the matrix is nonsingular, a reordering of the equations can be performed so that $a_{ii} \neq 0$. To speed up the convergence, the equations should be arranged so that a_{ii} is as large as possible.

Definition 2.6.1 (Residual Vector) *Suppose that $\tilde{\mathbf{x}} \in \mathbb{R}$ is an approximation to the solution of the linear system (2.30). The residual vector for $\tilde{\mathbf{x}}$ with respect this system is* $\mathbf{r} = \mathbf{b} - \mathbf{A}\tilde{\mathbf{x}}$.

In iterative methods, a residual vector is associated with each calculation of an approximation component to the solution vector. The objective of an iterative method is to generate a sequence of approximations that will cause the associated residual vectors to be reduced rapidly to zero. Let

$$\mathbf{r}_i^{(k)} = (r_{1i}^{(k)}, r_{2i}^{(k)}, \dots, r_{m_1i}^{(k)})^t$$

be the residual vector for the Gauss-Seidel method corresponding to the approximate solution vector $\mathbf{x}_i^{(k)}$ given by

$$\mathbf{x}_i^{(k)} = (x_1^{(k)}, x_2^{(k)}, \dots, x_{i-1}^{(k)}, x_i^{(k)}, \dots, x_{m_1}^{(k)})^t.$$

The mth component of $\mathbf{r}_i^{(k)}$ is

$$r_{mi}^{(k)} = b_m - \sum_{j=1}^{i-1} a_{mj}x_j^{(k)} - \sum_{j=i}^{m_1} a_{mj}x_j^{(k-1)}, \tag{2.37}$$

for each $m = 1, 2, \ldots, m_1$. In particular, for the ith component of $\mathbf{r}_i^{(k)}$ we have

$$a_{ii}x_i^{(k-1)} + r_{ii}^{(k)} = b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^{m_1} a_{ij}x_j^{(k-1)}. \tag{2.38}$$

Combining (2.35) and (2.38) we have

$$a_{ii}x_i^{(k-1)} + r_{ii}^{(k)} = a_{ii}x_i^{(k)}.$$

Consequently, the Gauss-Seidel method can be characterized as choosing $x_i^{(k)}$ to satisfy

$$x_i^{(k)} = x_i^{(k-1)} + \frac{r_{ii}^{(k)}}{a_{ii}}. \tag{2.39}$$

Modifying the Gauss-Seidel procedure given above in (2.39) to

$$x_i^{(k)} = x_i^{(k-1)} + \omega\frac{r_{ii}^{(k)}}{a_{ii}}. \tag{2.40}$$

for certain choices of positive ω reduces the norm of the residual vector and leads to significantly faster convergence. This type of methods defined in (2.40) are known as **relaxation methods.** If $0 < \omega < 1$, the methods are called **under-relaxation methods** and if $\omega > 1$, the techniques are known as **over-relaxation methods**, and are used to accelerate the convergence where Gauss-Seidel method is slow in convergence. These methods are abbreviated **SOR**, for Successive Over-Relaxation. Equation (2.40) can be written as

$$a_{ii}x_i^{(k)} + \omega\sum_{j=1}^{i-1} a_{ij}x_j^{(k)} = (1-\omega)a_{ii}x_i^{(k-1)} - \omega\sum_{j=i+1}^{m_1} a_{ij}x_j^{(k-1)} + \omega b_i \tag{2.41}$$

so in matrix form

$$(\mathbf{D} - \omega\mathbf{L})\mathbf{x}^{(k)} = [(1-\omega)\mathbf{D} + \omega\mathbf{U}]\mathbf{x}^{(k-1)} + \omega\mathbf{b} \tag{2.42}$$

or

$$\mathbf{x}^{(k)} = (\mathbf{D} - \omega\mathbf{L})^{-1}[(1-\omega)\mathbf{D} + \omega\mathbf{U}]\mathbf{x}^{(k-1)} + \omega(\mathbf{D} - \omega\mathbf{L})^{-1}\mathbf{b}. \tag{2.43}$$

The SOR technique can be express in the form

$$\mathbf{x}^{(k)} = \mathbf{T}_\omega\mathbf{x}^{(k-1)} + \mathbf{c}_\omega, \tag{2.44}$$

where $\mathbf{T}_\omega = (\mathbf{D} - \omega\mathbf{L})^{-1}[(1-\omega)\mathbf{D} + \omega\mathbf{U}]$ and $\omega(\mathbf{D} - \omega\mathbf{L})^{-1}\mathbf{b}$.

2.6.3 Block Iterative Methods

Assume that the vector $\mathbf{x}$ is partitioned into several disjoint sub-vectors (not necessarily of same size)

$$\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_s)^t.$$

Then $\mathbf{A}\mathbf{x} = \mathbf{b}$ can be written in the block form

$$\begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \ldots & \mathbf{A}_{1s} \\ \mathbf{A}_{21} & \mathbf{A}_{22} & \ldots & \mathbf{A}_{2s} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{A}_{s1} & \mathbf{A}_{s2} & \ldots & \mathbf{A}_{ss} \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_s \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \\ \mathbf{b}_s \end{bmatrix} \tag{2.45}$$

where the block $\mathbf{A}_{pq}$ is of size $n_p \times n_q$ (n_p being the size of $\mathbf{x}_p$) and the vector $\mathbf{b}$ is of size n_p. Assuming that the diagonal blocks are nonsingular the Jacobi and Gauss-Seidel methods can easily be extended to the block form. In the Block Jacobi method for $i = 1, 2, \ldots, s$

$$\mathbf{x}_i^{(k)} = \mathbf{A}_{ii}^{-1}\left(\sum_{\substack{j=1 \\ j\neq i}}^{s}(-\mathbf{A}_{ij}\mathbf{x}_j^{(k-1)}) + \mathbf{b}_i\right). \tag{2.46}$$

Similarly, the Gauss-Seidel method is given by

$$\mathbf{x}_i^{(k)} = \mathbf{A}_{ii}^{-1}\left(\mathbf{b}_i - \sum_{j=1}^{i-1}\mathbf{A}_{ij}\mathbf{x}_j^{(k)} - \sum_{j=i+1}^{s}\mathbf{A}_{ij}\mathbf{x}_j^{(k-1)}\right). \tag{2.47}$$

Obviously we now have to invert the matrix $\mathbf{A}_{ii}$ in order to update $\mathbf{x}_i$ and the larger the vectors $\mathbf{x}_i$ are, the more expensive each step of the method is likely to be, on the other hand the payoff may be faster convergence of the iterative method.

2.6.4 Convergence

All the iterative methods in this chapter define a sequence of iterates of the form

$$\mathbf{x}^{(k)} = \mathbf{T}\mathbf{x}^{(k-1)} + \mathbf{c}.$$

We now address the convergence properties and in particular sufficient conditions.

Lemma 2.6.1 *If the spectral radius $\rho(T) < 1$ then $(I-T)^{-1}$ exist, and*

$$(I-T)^{-1} = I + T + T^2 + \ldots$$

[56].

Proof. Let λ be an eigenvalue of T then $1-\lambda$ will be an eigenvalue of $I-T$, since if

$$T\mathbf{x} = \lambda\mathbf{x}$$
$$\Longrightarrow (I-T)\mathbf{x} = (1-\lambda)\mathbf{x}$$

for some eigenvector $\mathbf{x}$. But $|\lambda| \leqslant \rho(T) < 1$, so $\lambda = 1$ is not an eigenvalue of T, so 0 cannot be an eigenvalue of $I-T$. Hence $I-T$ is nonsingular. Hence $(I-T)^{-1}$ exists. For the second part

Let $S_m = I + T + T^2 + \ldots + T^m$. Then

$$(I-T)S_m = (I + T + T^2 + \ldots + T^m) - (T + T^2 + T^3 + \ldots + T^{m+1}) = I - T^{m+1}.$$

If $\rho(T) < 1$ then $\lim_{m\to\infty} T^m = 0$, so we have

$$\lim_{m\to\infty}(I-T)S_m = \lim_{m\to\infty}(I - T^{m+1}) = I$$

thus,

$$(I-T)^{-1} = \lim_{m\to\infty} S_m = I + T + T^2 + \ldots.$$

■

Theorem 2.6.1 *[56] For any* $\mathbf{x}^{(0)} \in \mathbb{R}^n$*, the sequence* $\{\mathbf{x}^{(k)}\}_{k=0}^{\infty}$ *defined by*

$$\mathbf{x}^{(k)} = T\mathbf{x}^{(k-1)} + \mathbf{c}, \quad for\, each\, k \geqslant 1, \tag{2.48}$$

converges to the unique solution of $\mathbf{x} = T\mathbf{x} + \mathbf{c}$ *if and only if* $\rho(T) < 1$.

Proof. First assume that $\rho(T) < 1$. *From equation (2.48) we have*

$$\begin{aligned} \mathbf{x}^{(k)} &= T\mathbf{x}^{(k-1)} + \mathbf{c} \\ &= T(T\mathbf{x}^{(k-2)} + \mathbf{c}) + \mathbf{c} \\ &= T^2\mathbf{x}^{(k-2)} + (T + I)\mathbf{c} \\ &\vdots \\ &= T^k\mathbf{x}^{(0)} + (T^{k-1} + \ldots + T + I)\mathbf{c}. \end{aligned}$$

Since $\rho(T) < 1$*, the matrix* T *is convergent*[7] *and*

$$\lim_{k\to\infty} T^k\mathbf{x}^{(0)} = 0$$

Using Lemma 2.6.1 we have

$$\lim_{k\to\infty} \mathbf{x}^{(k)} = \lim_{k\to\infty} T^k\mathbf{x}^{(0)} + \lim_{k\to\infty} \Big(\sum_{j=0}^{k-1} T^j\Big)\mathbf{c} = \mathbf{0} + (I - T)^{-1}\mathbf{c} = (I - T)^{-1}\mathbf{c}.$$

Hence the sequence $\{\mathbf{x}^{(k)}\}_{k=0}^{\infty}$ *converges to the unique solution* $\mathbf{x} = (I - T)^{-1}\mathbf{c}$ *as* $\mathbf{x} = T\mathbf{x} + \mathbf{c} \implies (I - T)\mathbf{x} = \mathbf{c}$.

Now consider the converse, let us assume that $\mathbf{x}$ *is the unique solution of the equation (2.48). Let* $\mathbf{c} = \mathbf{0}$ *then* $\mathbf{x}$ *is the solution of the equation* $\mathbf{x} = T\mathbf{x}$. *Let* $\mathbf{z} \in \mathbb{R}^n$ *be an arbitrary vector and write* $\mathbf{x}^{(0)} = \mathbf{x} - \mathbf{z}$*, we have*

$$\begin{aligned} \lim_{k\to\infty} T^k\mathbf{z} &= \lim_{k\to\infty} T^k(\mathbf{x} - \mathbf{x}^{(0)}) \\ &= \lim_{k\to\infty} T^{k-1}(T\mathbf{x} - T\mathbf{x}^{(0)}) \\ &= \lim_{k\to\infty} T^{k-1}(\mathbf{x} - \mathbf{x}^{(1)}) \\ &= \lim_{k\to\infty} T^{k-2}(\mathbf{x} - \mathbf{x}^{(2)}) \\ &\vdots \\ &= \lim_{k\to\infty} (\mathbf{x} - \mathbf{x}^{(k)}) = 0 \end{aligned}$$

Since $\mathbf{z} \in \mathbb{R}^{m_1}$ *is arbitrary, so* T *is convergent hence* $\rho(T) < 1$. ■

Definition 2.6.2 (Diagonally Dominant) *A square matrix* A *is said to be diagonally dominant if*

$$|a_{ii}| \geqslant \sum_{\substack{j=1 \\ j\neq i}}^{m_1} |a_{ij}|, \qquad i = 1, 2, \ldots, m_1.$$

[7] A square matrix A is said to be convergent if $\lim_{k\to\infty} A^k = 0$.

and is called strictly diagonally dominant if

$$|a_{ii}| > \sum_{\substack{j=1 \\ j\neq i}}^{m_1} |a_{ij}|, \qquad i = 1, 2, \dots, m_1.$$

Theorem 2.6.2 *If, for a system of equations (2.30), A is strictly diagonally dominant, the Gauss-Seidel iterations converge for any* $\mathbf{x}^{(0)}$. *[56]*

Proof. If A is a diagonally dominant matrix then

$$|a_{ii}| > \sum_{\substack{j=1 \\ j\neq i}}^{m_1} |a_{ij}|, \qquad i = 1, 2, \dots, m_1.$$

Since in Gauss-Seidel iterations we have

$$T_{GS} = (D - L)^{-1}U.$$

Let λ *be the dominant eigenvalue of the iteration matrix* T_{GS} *and let* $\mathbf{x}$ *be the corresponding eigenvector such that* $|x_m| = 1$ *and* $|x_j| \leqslant 1$ *for* $j \neq m$*, then*

$$(D - L)^{-1}U\mathbf{x} = \lambda\mathbf{x}$$

$$\Rightarrow U\mathbf{x} = \lambda(D - L)\mathbf{x} \tag{2.49}$$

$$\begin{aligned}
\Rightarrow \sum_{j<m} a_{mj}x_j &= \lambda(a_{mm}x_m - \sum_{j>m} a_{mj}x_j) \\
\Rightarrow \Big|\sum_{j<m} a_{mj}x_j\Big| &= |\lambda|\Big|(a_{mm}x_m - \sum_{j>m} a_{mj}x_j)\Big| \\
\Rightarrow |\lambda| &= \frac{|\sum_{j<m} a_{mj}x_j|}{|(a_{mm}x_m - \sum_{j>m} a_{mj}x_j)|} \\
&\leqslant \frac{\sum_{j<m} |a_{mj}x_j|}{|a_{mm}x_m| - \sum_{j>m} |a_{mj}x_j|} \\
&= \frac{\sum_{j<m} |a_{mj}||x_j|}{|a_{mm}||x_m| - \sum_{j>m} |a_{mj}||x_j|} \\
&\leqslant \frac{\sum_{j<m} |a_{mj}|}{|a_{mm}| - \sum_{j>m} |a_{mj}|} \qquad as\, |x_m| = 1\, and\, |x_j| \leqslant 1.
\end{aligned}$$

Let $\sigma_1 = \sum_{j>m} |a_{mj}|$, $\sigma_2 = \sum_{j<m} |a_{mj}|$ *and* $d = |a_{mm}|$. *So*

$$|\lambda| \leqslant \frac{\sigma_2}{\sigma_1 + d}. \tag{2.50}$$

Now d, σ_1 *and* σ_2 *are non-negative integers and also A is strictly diagonally dominant so* $d > \sigma_1 + \sigma_2$

$$\Rightarrow d - \sigma_1 > \sigma_2$$

Hence by (2.50) we can say that $|\lambda| < 1$

$$\Rightarrow \rho(T_{GS}) < 1$$

and so by Theorem 2.6.1 we have that Gauss-Seidel iterations will be convergent. ■

*N*ote: In similar way we can prove for Jacobi iterations.

If we have a system of equations $\mathbf{A}\mathbf{u} = \mathbf{f}$ arising from the discretization of a PDE using the finite difference method on a rectangular domain then the matrix $\mathbf{A}$ is likely to be well structured and sparse, which means storage of $\mathbf{A}$ will not usually be required. The updating of each entry of u will typically involve just a few other entries. Next we illustrate the implementation of Jacobi and Gauss-Seidel methods for the case of Poisson's equation:

$$\begin{cases} -\Delta u(x,y) = f(x,y) & (x,y) \in \Omega = (0,1) \times (0,1) \\ u(x,y) = f^{\Gamma}(x,y) & (x,y) \in \Gamma = \partial\Omega. \end{cases} \tag{2.51}$$

A discretized form of Poisson equation is

$$\begin{cases} -\Delta_h u_h(x,y) = f_h(x,y) & (x,y) \in \Omega_h, \\ u_h(x,y) = f_h^{\Gamma}(x,y) & (x,y) \in \Gamma_h = \partial\Omega_h. \end{cases} \tag{2.52}$$

Definition 2.6.3 (Stencil Notation) *A general 2-D stencil denoted by $[s_{\ell_1,\ell_2}]_h$ and is given by*

$$[s_{\ell_1,\ell_2}]_h = \begin{bmatrix} & \cdots & \cdots & \cdots & \\ \cdots & s_{-1,1} & s_{0,1} & s_{1,1} & \cdots \\ \cdots & s_{-1,0} & s_{0,0} & s_{1,0} & \cdots \\ \cdots & s_{-1,-1} & s_{0,-1} & s_{1,-1} & \cdots \\ & \cdots & \cdots & \cdots & \end{bmatrix}_h, \tag{2.53}$$

where $s_{\ell_1,\ell_2} \in \mathbb{R}$, this defines an operator on the set of grid functions by

$$[s_{\ell_1,\ell_2}]_h u_(x,y) = \sum_{\ell_1,\ell_2} s_{\ell_1,\ell_2} u_h(x + \ell_1 h_1, y + \ell_2 h_2). \tag{2.54}$$

Five-points and compact nine-point stencils are

$$\begin{bmatrix} & s_{0,1} & \\ s_{-1,0} & s_{0,0} & s_{1,0} \\ & s_{0,-1} & \end{bmatrix}_h \quad \begin{bmatrix} s_{-1,1} & s_{0,1} & s_{1,1} \\ s_{-1,0} & s_{0,0} & s_{1,0} \\ s_{-1,-1} & s_{0,-1} & s_{1,-1} \end{bmatrix}_h \tag{2.55}$$

Near boundary points the stencils may have to be modified [118].

Jacobi Method:

Using five-point stencil, in the weighted Jacobi Method if grid point (i,j) is not adjacent to the boundary, $u_{i,j}$ is updated according to the equation

$$u_{i,j}^k = (1-\omega)u_{i,j}^{k-1} + \omega\left[\frac{h^2 f_{i,j} + u_{i+1,j}^{k-1} + u_{i-1,j}^{k-1} + u_{i,j+1}^{k-1} + u_{i,j-1}^{k-1}}{4}\right]. \tag{2.56}$$

For points adjacent to the boundary, some modification will be required to (2.56).

Gauss-Seidel Method:

Two different ordering schemes (corresponding to two different ways of stacking $\mathbf{u}$ into a vector) for Gauss Seidel method will be discussed here.

Lexicographic Ordering:

A lexicographic ordering of the grid points involves ordering the points in increasing order from left to right and up the rows so that the approximation at the bottom left point $(1,1)$ is updated first followed by the approximation at the point $(2,1)$ and so on with the approximation at the top right point (m_l-1, m_l-1) updated last. A Gauss-Seidel scheme used with lexicographic ordering is denoted by GS-LEX and the entry u_h corresponding to grid point (i,j) (not adjacent to the boundary) is given by

$$u_{i,j}^k = \frac{h^2 f_{i,j} + u_{i-1,j}^k + u_{i,j-1}^k + u_{i+1,j}^{k-1} + u_{i,j+1}^{k-1}}{4}. \tag{2.57}$$

Red-Black Ordering:
When a red-black ordering of the grid points is used the grid is colored in a checkerboard fashion as shown in Figure 2.9, entries of u_h corresponding to the red points are updated first followed by entries of u_h corresponding to the black points. A Gauss Seidel scheme with red-black ordering of the grid points is denoted by GS-RB. Entries of u_h corresponding to red grid points are given by

$$u_{i,j}^k = \frac{h^2 f_{i,j} + u_{i-1,j}^{k-1} + u_{i,j-1}^{k-1} + u_{i+1,j}^{k-1} + u_{i,j+1}^{k-1}}{4} \tag{2.58}$$

and then entries corresponding to black points are updated by

$$u_{i,j}^k = \frac{h^2 f_{i,j} + u_{i-1,j}^k + u_{i,j-1}^k + u_{i+1,j}^k + u_{i,j+1}^k}{4} \tag{2.59}$$

Because a five point approximation to the PDE is being used, the updating of each entry associated with a red point involves only entries associated to black points and vice versa. This means that after each sweep of GS-RB the residual $r_h = f_h - L_h u_h$ is zero at the black points. When each red point is updated using only black points and vice-versa, GS-RB has an advantage over GS-LEX in terms of parallel computing since all the entries of u_h corresponding to red points can be computed in parallel followed by all entries of u_h corresponding to black points. Note that because points are updated in different orders, one step of GS-LEX will not produce an identical result to one step of GS-RB with the same initial guess.

Line Relaxation: If u_h is stacked into a vector $\mathbf{u}$ lexicographically and split $\mathbf{u}$ into (m_l-1) sub-vectors each of size (m_l-1) then the sub-vector $\mathbf{u}_\ell$ will contains all the values of u_h corresponding to row ℓ of the grid, hence performing a block Jacobi or Gauss-Seidel iteration on this block system is equivalent to relaxing a whole row of the grid collectively, this is known as x-line relaxation. Using block Gauss-Seidel, $\mathbf{u}_\ell$ is updated using the following equation

$$\mathbf{u}_\ell^k = \mathbf{A}_{\ell\ell}^{-1}(\mathbf{u}_{\ell-1}^k + \mathbf{u}_{\ell+1}^{k-1} + h^2\mathbf{f}_\ell), \tag{2.60}$$

where $\mathbf{A}_{\ell\ell}$ is a tridiagonal matrix with 4 on the diagonal and -1 on the off diagonals. If u_h is stacked along columns of the grid and the resulting vector partitioned as above, the block relaxation methods relax whole columns of the grid collectively, this is known as y-line relaxation. A sweep of an alternating line relaxation consists of an x-line relaxation sweep followed by a y-line relaxation sweep. A line analogue of the red-black point-wise relaxation for line Gauss-Seidel is the zebra line relaxation here either rows or columns of the grid are colored alternately white and black, then the white lines are relaxed followed by the black lines, in most cases the approximation at a point on a white line will depend only on other points on that line and points on the adjacent black lines, hence a parallel implementation of zebra line Gauss-Seidel will be possible.

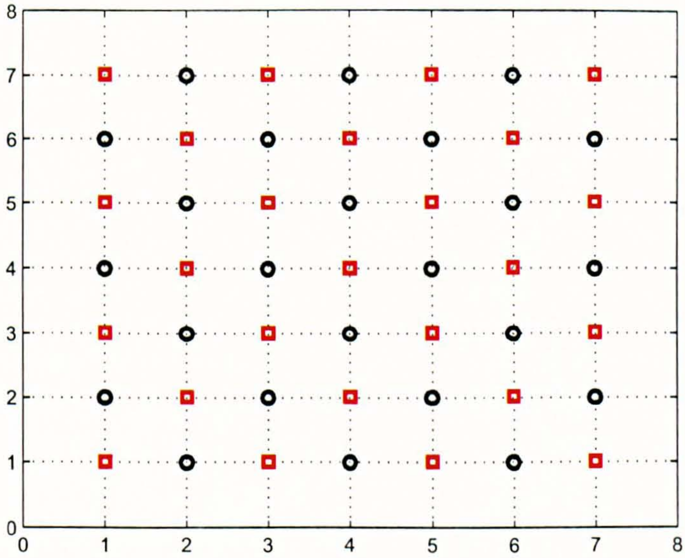


Figure 2.9: Red (□)-Black (○) ordering of grid points

In summary we have that Jacobi and Gauss-Seidel iterative techniques will be convergent if the spectral radius of their iterative matrices is less then 1 or equivalently we say that for a particular system $A\mathbf{x} = b$ if we want to use Jacobi and Gauss-Seidel iterative techniques, it is desirable that A must be diagonally dominant.

Even these iterative or relaxation methods may be very slow to converge, they are very fast for the smoothing error which is useful in multigrid methods see [39]. This fact can be revealed by a Fourier analysis, see Section 2.9.2.

2.7 Time Marching Schemes for Parabolic PDEs

In this section we explain the time marching methods used for parabolic equations, such as Explicit and Implicit difference methods and Additive Operator Splitting Scheme (AOS) and discuss their stability. The main references are [89], [98], [127] and [81].

2.7.1 Explicit Scheme (1-D)

Consider the 1-D Heat equation with constant coefficient

$$\frac{\partial u}{\partial t} = a\frac{\partial^2 u}{\partial x^2}, \qquad 0 \leqslant x \leqslant \pi,\ t \geqslant 0. \tag{2.61}$$

where $a > 0$ is constant and $u = u(x,t)$ and the boundary conditions are

$$\begin{cases} u(x,0) = \phi(x) & (\phi(x) \text{ given}) \\ u(0,t) = 0,\ u(\pi,t) = 0 & \text{for } t > 0. \end{cases}$$

The exact solution of this initial value problem (IVP) can be obtained by using Fourier series method with defining $\phi(x)$ as $-\phi(-x)$ for $-\pi \leqslant x \leqslant 0$, otherwise we use Fourier sine series, but here we use complex Fourier series i.e

$$u(x,t) = \sum_{m=-\infty}^{\infty} A_m \exp(imx - m^2 at) \tag{2.62}$$

where

$$A_m = \frac{1}{2\pi}\int_{-\pi}^{\pi} \phi(x)\exp(-\mathrm{i}mx)dx, \qquad \mathrm{i} = \sqrt{-1}. \tag{2.63}$$

Now we consider the discretized form of equation (2.61). Starting from $t_0 = 0$, $x_0 = 0$ and let Δx and Δt be the increments of the variables x and t respectively, with $\Delta x = \frac{\pi}{J}$ and $x_j = j\Delta x$ and $t_k = k\Delta t$ where $J \in Z$ and $j = 1, 2, \ldots, J$, $k = 0, 1, \ldots$. Also let $u_j^k = u(x_j, t_k)$, (x_j, t_k) is called the grid point. Then the explicit difference equation will be:

$$\frac{u_j^{k+1} - u_j^k}{\Delta t} = a\frac{u_{j+1}^k - 2u_j^k + u_{j-1}^k}{(\Delta x)^2}, \tag{2.64}$$

where $j = 1, 2, \ldots, J-1$, $k = 0, 1, \ldots$. The boundary conditions will be

$$u_0^k = 0, \quad u_J^k = 0, \quad k = 0, 1, \ldots \tag{2.65}$$

and the initial conditions will be

$$u_j^0 = \phi(j\Delta x), \quad j = 0, 1 \ldots, J. \tag{2.66}$$

These equations can be used recursively to determine all the u_j^k for $0 \leqslant j \leqslant J$ and $k \geqslant 0$.

2.7.2 Stability

Let $u(x,t)$ be the exact solution of the initial value problem and u_j^k is the solution of the finite difference equation, then the error approximation will be

$$u_j^k - u(j\Delta x, k\Delta t).$$

Then the following two questions arise:

- What is the behavior of $|u_j^k - u(j\Delta x, k\Delta t)|$ as $k \to \infty$ for fixed Δt and Δx ?
- What is the behavior of $|u_j^k - u(j\Delta x, k\Delta t)|$ as $\Delta x, \Delta t \to 0$ for fixed value of $t_k = k\Delta t$?

We note that an explicit solution of difference equation (2.64) can also be written as Fourier series, i.e we can find the solution of (2.64) of the form

$$u_j^k = \sum_{m=-\infty}^{\infty} A_m e^{\mathrm{i}mj\Delta x}(\xi(m))^k$$

where ξ and A_m are unknown. To find ξ and A_m we proceed as follows. Let us put $u_j^k = A_m \xi^k e^{\mathrm{i}mj\Delta x}$, $m \in Z$ in equation (2.64), to find the value of ξ we have

$$\begin{aligned} u_j^{k+1} &= u_j^k + \frac{a\Delta t}{(\Delta x)^2}\left(u_{j+1}^k - 2u_j^k + u_{j-1}^k\right) \\ \Rightarrow A_m\xi^{k+1}e^{\mathrm{i}mj\Delta x} &= A_m\xi^k e^{\mathrm{i}mj\Delta x} + \frac{a\Delta t}{(\Delta x)^2}\Big(A_m\xi^k e^{\mathrm{i}m(j+1)\Delta x} - 2A_m\xi^k e^{\mathrm{i}mj\Delta x} \\ &+ A_m\xi^k e^{\mathrm{i}m(j-1)\Delta x}\Big) \end{aligned}$$

dividing through out by $A_m\xi^k e^{imj\Delta x}$ we get

$$\begin{aligned}\xi &= 1+\frac{a\Delta t}{(\Delta x)^2}\left(e^{im\Delta x}-2+e^{-im\Delta x}\right)\\ &= 1-\frac{2a\Delta t}{(\Delta x)^2}\left(1-\frac{e^{im\Delta x}+e^{-im\Delta x}}{2}\right)\\ &= 1-\frac{2a\Delta t}{(\Delta x)^2}\left(1-\cos m\Delta x\right)\end{aligned}$$

Thus $u_j^k = A_m\xi^k e^{imj\Delta x}$ will satisfy the difference equation (2.64) if

$$\xi = 1-\frac{2a\Delta t}{(\Delta x)^2}\left(1-\cos m\Delta x\right) = \xi(m) \qquad (2.67)$$

where the value of A_m is same as that for the exact solution (2.63) i.e

$$A_m = \frac{1}{2\pi}\int_{-\pi}^{\pi}\phi(x)\exp(-imx)dx$$

therefore, the function

$$u_j^k = \sum_{m=-\infty}^{\infty} A_m e^{imj\Delta x}(\xi(m))^k \qquad (2.68)$$

gives the exact solution of the difference equation (2.64). Now we check whether this function satisfies the boundary condition and initial conditions given in (2.65) and (2.66) with A_m as mentioned above. Each term in the above Fourier series solution of the difference equation satisfies the difference equation and so is their sum. If we put $k=0$ then we have

$$u_j^0 = \sum_{m=-\infty}^{\infty} A_m e^{imj\Delta x},$$

which is the Fourier series for $\phi(j\Delta x)$ implies that initial condition is satisfied. In similar way we can check the boundary conditions.

Since (2.68) is the exact solution of the difference equation and that gives the values of function at mesh points i.e at $x = j\Delta x$, let us use a finite number of coefficients from above coefficients, the above function may be written in the following way as

$$u_j^k = \sum_{m=-J}^{J} B_m e^{imj\Delta x}[\xi(m)]^k,$$

where the B_m are to be obtained from the initial conditions regarded as $2J+1$ linear equations in the $2J+1$ unknowns B_{-J} to B_J (Here j runs from $-J$ to J as the initial functions has been extended into the interval $(-\pi, 0)$).

Let us compare the growth factor in both solutions. In the exact solution the growth factor is $e^{-m^2a\Delta t}$ and for the difference equation is $\xi(m)$. As

$$\xi(m) = 1 - m^2a\Delta t + \frac{1}{12}m^4a\Delta t(\Delta x)^2 - \ldots.$$

and

$$e^{-m^2a\Delta t} = 1 - m^2a\Delta t + \frac{1}{2}m^4a^2(\Delta t)^2 - \ldots.$$

It is found that the two growth factors agree through first order terms. The two growth factors can be made to agree, to any desired accuracy, by taking Δt and Δx sufficiently small, thus there is a hope that the solution of the equation (2.64) may be good approximation to the solution of the differential equation. Thus the stability condition is

$$\max_m |\xi(m)| \leqslant 1. \tag{2.69}$$

The error $u_j^k - u(j\Delta x, n\Delta t)$ remains bounded as $k \to \infty$ for fixed $\Delta x, \Delta t$, for a general initial function with absolutely convergent Fourier series if and only if (2.69) is satisfied. Since the true solution $u(x,t)$ is bounded as $t \to \infty$, the error will be bounded if u_j^k is bounded as $k \to \infty$.

$$|u_j^k| = \left| \sum_{m=-\infty}^{\infty} A_m e^{\mathrm{i}mj\Delta x}(\xi(m))^k \right| \leqslant \sum_{m=-\infty}^{\infty} |A_m||\xi(m)|^k, \quad |e^{\mathrm{i}mj\Delta x}| \leqslant 1$$

and also from (2.69) we have that $|\xi(m)| \leqslant 1, \quad \forall m$, so

$$|u_j^n| \leqslant \sum_{m=-\infty}^{\infty} |A_m|.$$

As $\phi(x)$ has absolutely convergent Fourier series, so

$$\sum_{m=-\infty}^{\infty} |A_m|$$

converges and so $|u_j^k|$ is bounded. For the present example, applying the stability condition (2.69), we have from (2.67) that the growth factor $\xi(m)$ is real $\forall m$ and never exceeds 1. The condition is that the greatest negative value of $\xi(m)$ be not less than -1, the greatest negative value is attained when $\cos m\Delta x = -1$, since $\Delta x = \frac{\pi}{J}$ and this happens when m is is an odd multiple of J, so the condition of $\xi(m) \geqslant -1$ for such value of m is that

$$\frac{2a\Delta t}{(\Delta x)^2} \leqslant 1 \quad \Rightarrow \Delta t \leqslant \frac{(\Delta x)^2}{2a},$$

which is the stability condition for the explicit schemes.

The Fourier series based stability, introduced by J. von Neumann is applied to a simple problem with constant coefficient here, but this can be applied to the differential equation with variable coefficients. This method can be applied to a wide variety of types of difference equations, where the more elementary methods fail. In summary, two points of view on stability have been presented. In the first view we can take $t \to \infty$ with fixed Δt or $\Delta t \to 0$ with fixed t, it is necessary to observe the restriction $\Delta t \leqslant \frac{(\Delta x)^2}{2a}$ to prevent errors, from becoming so amplified as to make the whole calculations disapproving. If the above condition is not satisfied, then the symptom of instability shows up in a relatively small number of iterations. Whenever we use finite difference methods for IVP, we must know the stability condition.

2.7.3 Implicit schemes (1-D)

Let $f(x)$ be any function of x, we denote by δf_j or $(\delta f)_j$ the central difference

$$\delta f_j = f((j + \frac{1}{2})\Delta x) - f((j - \frac{1}{2})\Delta x)$$

where j may be an integer or an integer plus $\frac{1}{2}$, and

$$\delta^2 f_j = f((j+1)\Delta x) - 2f(j\Delta x) + f((j-1)\Delta x).$$

With this notation, we wish to consider the following implicit difference system for 1-D heat equation (2.61):

$$\frac{u_j^{k+1}-u_j^k}{\Delta t}=a\frac{\theta(\delta^2u)_j^{k+1}+(1-\theta)(\delta^2u)_j^k}{(\Delta x)} \tag{2.70}$$

where $0\leqslant\theta\leqslant 1$ is a real constant (when $\theta=0$, the system reduces to (2.64)).

2.7.4 Stability

Again by putting $u_j^k=A_me^{imj\Delta x}\xi^k$ into (2.70) and simplifying we have

$$\xi(m)=\frac{1-(1-\theta)L(1-\cos m\Delta x)}{1+\theta L(1-\cos m\Delta x)} \tag{2.71}$$

where $L=\dfrac{2a\Delta t}{(\Delta x)^2}$. For stability we need to check the condition $|\xi(m)|\leqslant 1$. The quantity $\xi(m)$ should be real for all m, and never exceeds 1. In Figure 2.10 the values of $\xi(m)$ are plotted as a function of the argument $y=L(1-\cos m\Delta x)$ for various values of θ. As y increases through positive values, the value of $\xi(m)$ falls monotonically from 1 to $-\dfrac{(1-\theta)}{\theta}$. If $1/2\leqslant\theta\leqslant 1$, the asymptote is not less than -1, as in Fig 2.10. The graph with $\triangleright's$ and $\square's$ shows this, hence the difference equations (2.70) are always stable. But if $0\leqslant\theta<1/2$, y must be restricted, for stability, by the value at which the curve intersects the line $\xi=-1$, in Fig 2.10. The graph with $\circ's$ shows this, hence the stability condition must be

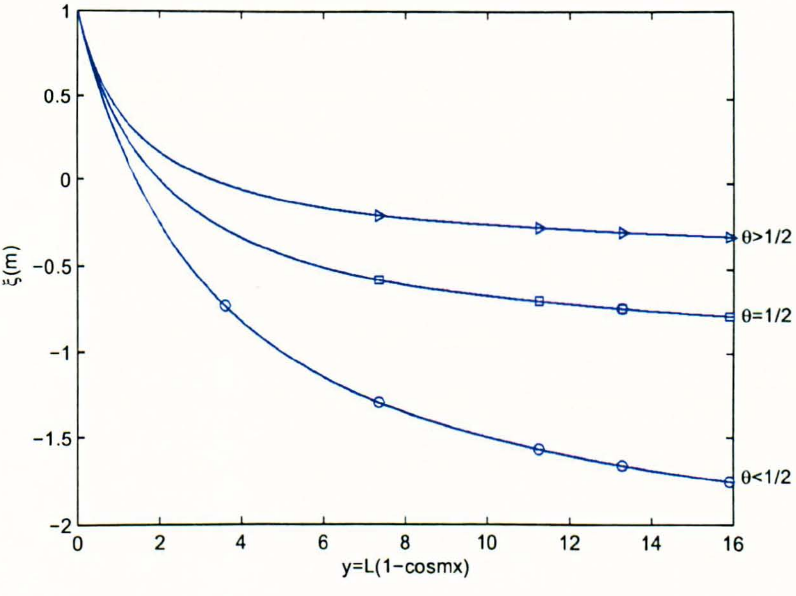


Figure 2.10: Growth factor for the equation (2.70), through (2.71)

$$\begin{cases}\frac{2a\Delta t}{(\Delta x)^2}\leqslant\frac{1}{1-2\theta} & \text{if } 0\leqslant\theta<1/2\\ \text{always stable} & \text{if } 1/2\leqslant\theta\leqslant 1\end{cases}$$

or

$$\begin{cases} \Delta t \leqslant \frac{(\Delta x)^2}{2a(1-2\theta)} & \text{if } 0 \leqslant \theta < 1/2 \\ \text{always stable} & \text{if } 1/2 \leqslant \theta \leqslant 1. \end{cases}$$

In the next section we discuss the 2-D case of the heat equation.

2.7.5 Stability in 2-D

Let us consider the 2-dimensional heat equation

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}. \tag{2.72}$$

Let

$$u_{jl}^k = u(j\Delta x, l\Delta y, k\Delta t)$$

and

$$\phi_{jl}^k = \frac{1}{(\Delta x)^2}\left(u_{j+1,l}^k - 2u_{jl}^k + u_{j-1,l}^k\right) + \frac{1}{(\Delta y)^2}\left(u_{j,l+1}^k - 2u_{jl}^k + u_{j,l-1}^k\right) \tag{2.73}$$

Then the finite difference approximation for (2.72), is the following

$$\frac{u_{jl}^{k+1} - u_{jl}^k}{\Delta t} = \theta\phi_{jl}^{k+1} + (1-\theta)\phi_{jl}^k. \tag{2.74}$$

If $0 < \theta < 1$ then (2.74) is an semi-implicit, if $\theta = 0$, then it reduces to explicit and if $\theta = 1$ it reduces to implicit. Putting $u_{jl}^k = exp(\mathrm{i}k_1 j\Delta x + \mathrm{i}k_2 l\Delta y)\xi^k$ in (2.73) we have

$$\begin{aligned}
\phi_{jl}^k &= \frac{1}{(\Delta x)^2}\Big(exp(\mathrm{i}k_1(j+1)\Delta x + \mathrm{i}k_2 l\Delta y) \\
&\quad -2exp(\mathrm{i}k_1 j\Delta x + \mathrm{i}k_2 l\Delta y) + exp(\mathrm{i}k_1(j-1)\Delta x + \mathrm{i}k_2 l\Delta y)\Big)\xi^k \\
&\quad +\frac{1}{(\Delta y)^2}\Big(exp(\mathrm{i}k_1 j\Delta x + \mathrm{i}k_2(l+1)\Delta y) \\
&\quad -2exp(\mathrm{i}k_1 j\Delta x + \mathrm{i}k_2 l\Delta y) + exp(\mathrm{i}k_1 j\Delta x + \mathrm{i}k_2(l-1)\Delta y)\Big)\xi^k \\
&= -2\xi^k exp(\mathrm{i}k_1 j\Delta x + \mathrm{i}k_2 l\Delta y)\left[\frac{1}{(\Delta x)^2}(1-\cos k_1\Delta x) + \frac{1}{(\Delta y)^2}(1-\cos k_2\Delta y)\right] \\
\Delta t\phi_{ij}^k &= -2\xi^k exp(\mathrm{i}k_1 j\Delta x + \mathrm{i}k_2\Delta y)\left[\frac{\Delta t}{(\Delta x)^2}(1-\cos k_1\Delta x) + \frac{\Delta t}{(\Delta y)^2}(1-\cos k_2\Delta y)\right] \\
\Rightarrow \Delta t\phi_{jl}^k &= exp(\mathrm{i}k_1 j\Delta x + \mathrm{i}k_2 j\Delta y)\Psi\xi^k \\
\text{where}\quad \Psi &= -2\left[\frac{\Delta t}{(\Delta x)^2}(1-\cos k_1\Delta x) + \frac{\Delta t}{(\Delta y)^2}(1-\cos k_2\Delta y)\right]
\end{aligned}$$

Now equation (2.74) can be written as

$$u_{jl}^{k+1} - u_{jl}^k = \theta\Delta t\phi_{jl}^{k+1} + (1-\theta)\Delta t\phi_{jl}^k$$

and putting the expression for ϕ_{jl}^k and ϕ_{jl}^{k+1} we have

$$\begin{aligned}
&exp(\mathrm{i}k_1 j\Delta x + \mathrm{i}k_2 j\Delta y)\xi^{k+1} - exp(\mathrm{i}k_1 j\Delta x + ik_2 j\Delta y)\xi^k \\
&= \theta exp(\mathrm{i}k_1 j\Delta x + ik_2 j\Delta y)\Psi\xi^{k+1} + (1-\theta)exp(\mathrm{i}k_1 j\Delta x + ik_2 j\Delta y)\Psi\xi^k
\end{aligned}$$

$$\begin{aligned}\Rightarrow \xi - 1 &= \theta\Psi\xi + (1-\theta)\Psi \\ \Rightarrow \xi &= \frac{1+(1-\theta)\Psi}{1-\theta\Psi}. \qquad (2.75)\end{aligned}$$

This is the amplification term. Now requiring condition $|\xi| \leqslant 1$ we need to find the bounds for Ψ, i.e the maximum and minimum value of Ψ as a function of two real variables $k_1\Delta x$ and $k_2\Delta y$. This function is periodic and analytic so the extreme values occurs when

$$\frac{\partial\Psi}{\partial(k_1\Delta x)} = \frac{\partial\Psi}{\partial(k_2\Delta y)} = 0$$

and this is only possible when

$$\sin k_1\Delta x = \sin k_2\Delta y = 0 \Rightarrow \cos k_1\Delta x = \pm 1 \quad and \quad \cos k_2\Delta y = \pm 1$$

so

$$-4\Big[\frac{\Delta t}{(\Delta x)^2} + \frac{\Delta t}{(\Delta y)^2}\Big] \leqslant \Psi \leqslant 0.$$

From equation (2.75) we have

- ξ is real.
- $\xi \leqslant 1$ (always)
- If $\theta \geqslant \frac{1}{2}$ we have $\xi \geqslant -1$, so stability is achieved.
- If $0 \leqslant \theta < \frac{1}{2}$, a stability requirement is $\xi \geqslant -1$, which imposes a restriction on Δt, thus we have to find this stability condition.

As

$$\begin{aligned}\xi \geqslant -1 \quad i.e \quad \frac{1+(1-\theta)\Psi}{1-\theta\Psi} &\geqslant -1 \\ \Rightarrow 1 + (1-\theta)\Psi &\geqslant -1 + \theta\Psi \\ \Rightarrow \Psi - 2\theta\Psi \geqslant -2 \Rightarrow \Psi &\geqslant \frac{-2}{1-2\theta} \\ \Rightarrow -4\Big[\frac{\Delta t}{(\Delta x)^2} + \frac{\Delta t}{(\Delta y)^2}\Big] \geqslant \frac{-2}{1-2\theta} \Rightarrow \frac{\Delta t}{(\Delta x)^2} + \frac{\Delta t}{(\Delta y)^2} &\leqslant \frac{1}{2-4\theta}.\end{aligned}$$

Thus we have the following stability condition

$$\begin{cases} \frac{\Delta t}{(\Delta x)^2} + \frac{\Delta t}{(\Delta y)^2} \leqslant \frac{1}{2-4\theta} & \text{if } 0 \leqslant \theta < 1/2 \\ \text{always stable} & \text{if } 1/2 \leqslant \theta \leqslant 1. \end{cases}$$

2.7.6 Additive Operator Splitting (AOS) Scheme

This scheme was introduced by Tai [81] in 1992 and Weickert [127] in 1998. This scheme applies to PDEs of the form

$$\begin{aligned} u_t &= div(\tilde{g}\nabla u) + \mathrm{r}_1(\mathbf{x}, u) \\ &= (\tilde{g}u_{x_1})_{x_1} + \ldots + (\tilde{g}u_{x_d})_{x_d} + \mathrm{r}_1(\mathbf{x}, u) \qquad (2.76)\end{aligned}$$

in $[0,T] \times \Omega \subset \mathbf{R}^d$ with initial and boundary conditions

$$u(0,.) = u_0 \quad and \quad \frac{\partial u}{\partial n} = 0 \quad on\, \partial\Omega.$$

Here $\widetilde{g}$ is the diffusivity function and r denotes a reaction term. Let us assume that the diffusivity function depends upon only on the location, i.e $\widetilde{g} = \widetilde{g}(\mathrm{x})$. Without loss of generality we drop the reaction term r_1.

A backward Euler implicit step for the time discretization and a spatial finite difference discretization yields the semi-implicit evolution scheme

$$u^{k+1} = \left(I_{N_d} - \frac{\Delta t}{h^2}A\right)^{-1} u^k, \qquad k = 1, 2, \ldots, \tag{2.77}$$

where u^k is represented by a column vector of length N^d. The matrix A is of size $N^d \times N^d$, it can be represented in tensor product notation as

$$A = A_1 \otimes A_2 \otimes \ldots \otimes A_d,$$

where A_i is the tridiagonal matrix corresponding to finite difference discretization of the partial differential operators $(\widetilde{g}u_{x_i})_{x_i}$. Let us define the notation $\otimes$: let

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \quad and \quad B = \begin{pmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{pmatrix}$$

then

$$A \otimes B = \begin{pmatrix} a_{11}B & a_{12}B & a_{13}B \\ a_{21}B & a_{22}B & a_{23}B \\ a_{31}B & a_{32}B & a_{33}B \end{pmatrix}$$

Since A contains $2d+1$ nonzero diagonals and has band width dN, the linear systems in each iteration steps are more difficult to solve than in one-dimensional case where the iteration matrices are tridiagonal and the *Thomas algorithm* does the job [127].

The AOS scheme provides an additive decomposition of the evolution matrix. The scheme proposes

$$u^{k+1} = \frac{1}{d}\sum_{i=1}^{d}\left(I_{N_i} - d\frac{\Delta t}{h^2}A_i\right)^{-1} u^k, \qquad k = 1, 2, \ldots. \tag{2.78}$$

In words, each iteration step requires the old iterate to be propagated in all coordinate direction separately. Then the new iteration is given by the average of these intermediate solutions. It is stated without proof in [127] that the AOS scheme is an $\mathcal{O}(\Delta t) + \mathcal{O}(h^2)$ accurate finite difference approximation to the original equation, hence equally accurate as the usual explicit and implicit schemes. And a proof is given in [89].

Theorem 2.7.1 *The AOS scheme (2.78) with d=2 corresponding to the finite difference equation*

$$(\mathcal{F}\phi)^k := \frac{1}{\Delta t}\phi^{k+1} - \frac{1}{2\Delta t}\left(I - 2\frac{\Delta t}{h^2}A_1\right)^{-1}\phi^k - \frac{1}{2\Delta t}\left(I - 2\frac{\Delta t}{h^2}A_2\right)^{-1}\phi^k = 0,$$

$k = 0, 1, \ldots,$ *is consistent in* l_∞*-norm of first order in time and second order in space with the PDE (2.76).*

Proof. Let ϕ be an arbitrary smooth function, for which we will examine the local truncation error

$$T_{ij}^k(\phi) = (\mathcal{F}u)_{ij}^k - L(\phi)(t_k, x_{ij}),$$

where L is the spatial partial differential operator given by

$$L(\phi) = div(\tilde{g}\nabla\phi) = (\tilde{g}\phi_x)_x + (\tilde{g}\phi_y)_y.$$

Besides the function ϕ, we will need in each time step the function ν solving

$$\nu_t - 2(\tilde{g}\nu_x)_x = 0 \qquad \text{with initial condition} \quad \nu(t_k,.) = \phi(t_k,.) \tag{2.79}$$

and the function ω solving

$$\omega_t - 2(\tilde{g}\omega_y)_y = 0 \qquad \text{with initial condition} \quad \omega(t_k,.) = \phi(t_k,.) \tag{2.80}$$

Both PDEs are supplied with usual Neumann boundary conditions. These equations are one-dimensional diffusion equations, which are accelerated by the factor 2 compared to the PDE (2.76). For ease of notation, we do not index ν and ω by time step k, even though it should be noted that these functions are different in each time step k (since the initial condition of their PDEs differ in each step). Let r be the function such that

$$\phi = \frac{1}{2}\nu + \frac{1}{2}\omega + r \qquad in\,[t_k, t_{k+1}] \times \Omega$$

at each time step n. For the finite difference operator at time step k, we obtain

$$\begin{aligned}(\mathcal{F}\phi)^k &= \underbrace{\frac{1}{2\Delta t}\mathrm{v}^{k+1} - \frac{1}{2\Delta t}\Big(I - 2\frac{\Delta t}{h^2}A_1\Big)^{-1}\phi^k}_{F_1} \\ &+ \underbrace{\frac{1}{2\Delta t}\mathrm{w}^{k+1} - \frac{1}{2\Delta t}\Big(I - 2\frac{\Delta t}{h^2}A_2\Big)^{-1}\phi^k}_{F_2} + \frac{1}{\Delta t}\mathrm{r}^{k+1}\end{aligned}$$

where ϕ^k, ϕ^{k+1}, v^{k+1}, w^{k+1}, r^{k+1}denote the vectors (reordered already by rows) containing the values $u, v\, and\, w$ on the spatial grid point at time steps k and $k+1$ respectively. Consider F_1

$$\frac{1}{2\Delta t}\mathrm{v}^{k+1} - \frac{1}{2\Delta t}\Big(I - 2\frac{\Delta t}{h^2}A_1\Big)^{-1}\phi^k$$

which apart from the factor $\frac{1}{2}$ corresponds to the very first step of an implicit scheme for the PDE (2.79). Since ν is initialized to $\phi(t_k,.)$ at time step k, this term is nothing else but the finite difference operator of the implicit scheme evaluated at the function ν. But ν is also the solution of this PDE with initial data ϕ^k, so that the local truncation error for the implicit scheme is of first order in time and second order in space i.e

$$\left\|\frac{1}{2\Delta t}\mathrm{v}^{k+1} - \frac{1}{2\Delta t}\Big(I - 2\frac{\Delta t}{h^2}A_1\Big)^{-1}\mathrm{u}^k\right\|_\infty = \mathcal{O}(\Delta t) + \mathcal{O}(h^2).$$

Of course, the same argument also applies to F_1, using the fact that ω solves the PDE (2.80). So we obtain

$$(\mathcal{F}\phi)^k_{ij} = \frac{1}{\Delta t}\mathrm{r}^{k+1}_{ij} + \mathcal{O}(\Delta t) + \mathcal{O}(h^2), \tag{2.81}$$

where the constants in the order terms do not depend on i, j or k.

It remains to take a look at r^{k+1}_{ij}:

$$\mathrm{r}^{k+1}_{ij} = \phi^{k+1}_{ij} - \frac{1}{2}\mathrm{v}^{k+1}_{ij} - \frac{1}{2}\mathrm{w}^{k+1}_{ij}.$$

We use the Taylor expansion of u, ν and ω at time k to give

$$\begin{aligned} \mathbf{r}_{ij}^{k+1} &= \phi_{ij}^{k} + \Delta t \phi_t(t_k, x_{ij}) + \mathcal{O}((\Delta t)^2) \\ &- \frac{1}{2}(\phi_{ij}^{k} + \Delta t \nu_t(t_k, x_{ij}) + \mathcal{O}((\Delta t)^2) \\ &- \frac{1}{2}(\phi_{ij}^{k} + \Delta t \omega_t(t_k, x_{ij}) + \mathcal{O}((\Delta t)^2) \\ &= \tau\Big(\phi_t(t_k, x_{ij}) - \frac{1}{2}\nu_t(t_k, x_{ij}) - \frac{1}{2}\omega_t(t_k, x_{ij})\Big) + \mathcal{O}((\Delta t)^2), \end{aligned}$$

We shall put this into the expression (2.81) for the FD operator. Furthermore, we exploit the facts that ν and ω satisfy diffusion equations and that at time t_k, we have $\phi = \nu = \omega$. Hence, $(\tilde{g}\phi_x)_x = (\tilde{g}\nu_x)_x$ and $(\tilde{g}\phi_y)_y = (\tilde{g}\omega_y)_y$ and we obtain from (2.81) and the above

$$\begin{aligned} (\mathcal{F}\phi)_{ij}^{k} &= \phi_t(t_k, x_{ij}) - \frac{1}{2}.2(\tilde{g}\phi_x)_x(t_k, x_{ij}) \\ &- \frac{1}{2}.2(\tilde{g}\phi_y)_y(t_k, x_{ij}) + \mathcal{O}(\Delta t) + \mathcal{O}(h^2) \\ &= \phi_t(t_k, x_{ij}) - (div(\tilde{g}\phi))(t_k, x_{ij}) + \mathcal{O}(\Delta t) + \mathcal{O}(h^2) \end{aligned}$$

Putting this into the local truncation error yields the required consistency

$$T_{ij}^{k}(\phi) = \mathcal{O}(\Delta t) + \mathcal{O}(h^2).$$

The corresponding assertion for the l_∞-norm follows from the fact that the constants in the order terms do not depend on time or location. Finally, we mention that the ghost value technique inherent in the one-dimensional implicit FD schemes ensures the second-order accuracy of the scheme with respect to the Neumann boundary conditions. ■

More details can be found in [89].

In summary, the explicit method is computationally very cheap. The computational and storage effort is linear in the number of pixels N. Since for explicit scheme we have $\Delta t < \frac{(\Delta x)^2}{2a}$, this is often a very severe step size restriction. It means that the use of an explicit scheme is limited rather by its stability than its accuracy. In one dimension, the semi-implicit scheme requires to solve a diagonally dominant tridiagonal system of equations, and this is an unconditionally stable and we can get a desired accuracy without the need to choose small time steps for stability. In higher dimension the semi-implicit scheme remains absolutely stable but it is so laborious to solve the system of equations obtained from the discretization. Additive operator splitting schemes (AOS) split the m-dimensional spatial operator into a sum of m one dimensional space discretizations. The update of each grid point involves only two neighbors in each dimension, thus reducing the system to a set of tridiagonal system, which is similar to semi-implicit scheme in one dimension.

2.8 Basic Multigrid Methods

Practical multigrid methods were first introduced in the 1970s by Brandt [14]. These methods can solve elliptic PDEs discretized on $N = m_1 \times m_2$ grid points in $\mathcal{O}(N)$ operations. The multigrid methods can solve general elliptic equations with nonconstant coefficients with hardly any loss in efficiency, even nonlinear equations can be solved with comparable speed. There is no single multigrid algorithm that solves all elliptic problems.

Rather there is a multigrid technique that provides the framework for solving these problems. To solve your own problem, one needs to adjust the various components of the algorithm within this framework. In this section we give some introduction of basic ideas behind the linear and nonlinear multigrid methods. To explain the various components of multigrid methods, we will consider the classic example of Poisson's equation on the unit square domain. See : [14, 39, 68, 118, 97, 128] and many references therein for more details.

2.9 Multigrid Method

Simple iterative methods (such as the Jacobi method and Gauss Seidel method) tend to damp out high frequency components of the error fastest. This has led people to develop methods based on the following heuristic:

1. Perform some steps of a basic method in order to smooth out the error.
2. Restrict the current state of the problem to a subset of the grid points, the so-called "coarse grid", and solve the resulting projected problem.
3. Interpolate the coarse grid solution back to the original grid, and perform a number of steps of the basic method again.

Steps 1 and 3 are called "pre-smoothing" and "post-smoothing" respectively; by applying this method recursively to step 2 it becomes a true "multigrid" method. Usually the generation of subsequently coarser grids is halted at a point where the number of variables becomes small enough that direct solution of the linear system is feasible. The method outlined above is said to be a "V-cycle" method, since it descends through a sequence of subsequently coarser grids, and then ascends this sequence in reverse order. A "W-cycle" method results from visiting the coarse grid twice, with possibly some smoothing steps in between.

An analysis of multigrid methods is relatively straightforward in the case of simple differential operators such as the Poisson operator on tensor product grids. In that case, each next coarse grid is taken to have the double grid spacing of the previous grid. In two dimensions, a coarse grid will have one quarter of the number of points of the corresponding fine grid. Since the coarse grid is again a tensor product grid, a Fourier analysis can be used. For the more general case of self-adjoint elliptic operators on arbitrary domains a more sophisticated analysis is needed. Many multigrid methods can be shown to have an (almost) optimal number of operations $\mathcal{O}(N)$, that is, the work involved is proportional to the number of variables N.

2.9.1 Elements of the Multigrid Method

The two main ingredients of multigrid method are *error smoothing* and *coarse grid correction.*

Error Smoothing:-

Many basic relaxation schemes like Jacobi and Gauss Seidel methods when used to solve elliptic PDEs, discretized on Cartesian grids are slow to converge because of low frequency error components, however they do posses a smoothing property. These schemes are effective at removing the oscillatory Fourier modes of the error in an approximation but may not effective at removing the smooth[8] modes of the error i.e they smooth the

[8] A function is called smooth if its Fourier coefficients are decaying, i.e it is essentially in a span of some low frequency Fourier basis functions.

error while not necessarily reducing its size greatly. A smooth quantity can however be well approximated on a coarser grid.

Coarse grid correction:-

Let us consider a linear system

$$\mathcal{L}u = f. \tag{2.82}$$

Let v be an approximation to the solution u then the error e is given by

$$e = u - v,$$

which give rise to a residual equation

$$\mathcal{L}e = f - \mathcal{L}v = r, \tag{2.83}$$

where r is the residual. This residual equation is used to approximate the error e. Of course this equation is as expensive as the original one (2.82), but if we replace $\mathcal{L}$ by some simple approximation $\hat{\mathcal{L}}$ (for Jacobi method diagonal D of $\mathcal{L}$ is used for approximation) an approximation of the error can be found relatively cheaply, then used to correct v. Then repeat the process until convergence. The following example will be used in subsequent discussion

Example 2.9.1 *Let us consider the Poisson equation*

$$\begin{cases} -\Delta u(x,y) = f(x,y) & (x,y) \in \Omega = (0,1) \times (0,1), \\ u(x,y) = f^{\Gamma}(x,y) & (x,y) \in \Gamma = \partial\Omega. \end{cases} \tag{2.84}$$

A discretized form of Poisson equation is

$$\begin{cases} -\Delta_h u_h(x,y) = f_h(x,y) & (x,y) \in \Omega_h \\ u_h(x,y) = f_h^{\Gamma}(x,y) & (x,y) \in \Gamma_h = \partial\Omega_h, \end{cases} \tag{2.85}$$

with $h = 1/m_1, m_1 \in \mathbb{N}$. The iteration formula of the classical lexicographical Gauss-Seidel method for Poisson equation is

$$\begin{aligned} u_h^{k+1}(x_i,y_j) &= \frac{1}{4}[h^2 f_h(x_i,y_j) + u_h^{k+1}(x_i - h, y_j) + u_h^k(x_i + h, y_j) \\ &+ u_h^{k+1}(x_i, y_j - h) + u_h^k(x_i, y_j + h)], \end{aligned} \tag{2.86}$$

where$(x_i,y_j) \in \Omega_h$ and u_h^k and u_h^{k+1} are the approximations of $u_h(x_i,y_j)$ before and after iteration, respectively. Consider the error

$$e_h^k(x_i,y_j) = u_h(x_i,y_j) - u_h^k(x_i,y_j).$$

Then (2.86) becomes

$$\begin{aligned} e_h^{k+1}(x_i,y_j) &= \frac{1}{4}[e_h^{k+1}(x_i - h, y_j) + e_h^k(x_i + h, y_j) \\ &+ e_h^{k+1}(x_i, y_j - h) + e_h^k(x_i, y_j + h)]. \end{aligned} \tag{2.87}$$

Error smoothing is one of the two basic principles of the multigrid approach. Many classical iterative methods like Gauss-Seidel etc. if appropriately applied to discrete elliptic problems have a strong smoothing effect on the error of any approximation. A quantity that is smooth on a certain grid can also be approximated on a coarse grid (a grid with double size) without any essential loss of information. In other words the error become smooth after few iterations, then we can approximate this error on a coarse grid with fewer grid points. Thus we can say that coarse grid procedure is less expensive then a fine grid procedure.

2.9.2 Smoothing Analysis

Definition 2.9.1 (Visible) *A component $e^{\mathrm{i}\theta x/h}$ is visible on Ω_h if there is no frequency θ_0 with $|\theta_0| < |\theta|$ such that*

$$e^{\mathrm{i}\theta_0 x/h} = e^{\mathrm{i}\theta x/h} \qquad \textit{for all } x \in \Omega_h.$$

A visible component does not coincide with any lower frequency on fine grid Ω_h.

1D case:
On fine grid Ω_h only Fourier components $e^{\mathrm{i}\theta x/h}$ with $\theta \in (-\pi, \pi]$ are "visible"
In local mode analysis the notion of low- and high-frequency components on the grid Ω_h is related to a coarse grid Ω_{2h}. In this way $e^{\mathrm{i}\theta x/h}$ on Ω_h is said to be an high-frequency component, with respect to the grid Ω_{2h}, if its restriction to Ω_{2h} is not visible there. Usually the high frequencies are those with $\frac{\pi}{2} \leqslant |\theta| \leqslant \pi$.
2D case:
Let us consider the vertex centered discretization of the domain Ω_h, let h_1, h_2 in x- and y- direction spatial step sizes respectively, then

$$\Omega_h = \{(\ell_1 h_1, \ell_2 h_2) : \ell_1, \ell_2 \in \mathbb{N}\}$$

$\theta = (\theta_1, \theta_2)$ is the "frequency". On fine grid Ω_h, only Fourier components

$$e^{\mathrm{i}\theta\mathbf{x}/\mathbf{h}} := e^{\mathrm{i}\theta_1 x/h_1} e^{\mathrm{i}\theta_2 y/h_2}, \qquad \mathbf{h} = (h_1, h_2)$$

with $|\theta| := \max(|\theta_1|, |\theta_2|) \leqslant \pi$ are "visible". The same idea can be extended to higher dimension.

We will consider 2D problem onwards, stencil notation of $\Delta_h u_h(\mathbf{x})$ is given by

$$\Delta_h u_h(\mathbf{x}) = \underbrace{\frac{1}{h^2}\begin{bmatrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{bmatrix}}_{L_h} u_h(\mathbf{x}),$$

where $\mathbf{x} = (x, y) \in \Omega_h$, $\mathbf{i} = \sqrt{-1}$ and $h_1 = h_2 = h$. Applying a difference operator L^h to Fourier components

$$L_h u_h e^{\mathrm{i}\theta\mathbf{x}/h} = \underbrace{\Big(\sum_{\mu_1 \in I}\sum_{\mu_2 \in I} a_{\mu_1\mu_2} e^{\mathrm{i}\theta_1\mu_1} e^{\mathrm{i}\theta_2\mu_2}\Big)}_{\widetilde{L}_h(\theta)} . e^{\mathrm{i}\theta\mathbf{x}/h}, \tag{2.88}$$

$\widetilde{L}_h(\theta)$ is called the eigenvalue of L_h, and $e^{i\theta\mathbf{x}/h}$ are eigenfunctions. For example, let $\theta = (\theta_1, \theta_2)$ and

$$L_h = \Delta_h = \frac{1}{h^2}\begin{bmatrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{bmatrix}$$

$$\widetilde{L}_h(\theta) = \frac{2}{h^2}(\cos\theta_1 + \cos\theta_2 - 2). \tag{2.89}$$

Example 2.9.2 *Let us consider the linear problem*

$$L_h U_h(\mathbf{x}) = f_h \qquad \textit{on } \Omega_h$$

where L_h is a difference operator, this problem is based on an operator splitting

$$L_h = A_h + B_h$$

where A_h and B_h are again difference operators. Let u_h be the initial approximation which produces a new approximation $\bar{u}_h$ by solving

$$A_h u_h(\mathbf{x}) + B_h \bar{u}_h(\mathbf{x}) = f_h(\mathbf{x})$$

at each grid point $\mathbf{x} \in \Omega_h$. Let $e_h = U_h - u_h$ and $\bar{e}_h = U_h - \bar{u}_h$ be the error before and after relaxation sweep[9] respectively, then we have the following error equation

$$A_h e_h(\mathbf{x}) + B_h \bar{e}_h(\mathbf{x}) = 0$$

at each grid point $\mathbf{x} \in \Omega_h$. Also let $e_h = \mathcal{A}e_{\mathrm{i}\theta\mathbf{x}/h}$ and $\bar{e}_h = \bar{\mathcal{A}}e^{\mathrm{i}\theta\mathbf{x}/h}$, $\mathcal{A}$, $\bar{\mathcal{A}} \in \mathbb{R}$ are the error amplitudes before and after relaxation respectively, then error equation becomes

$$\widetilde{A}_h(\theta)\mathcal{A} + \widetilde{B}_h(\theta)\bar{\mathcal{A}} = 0$$
$$\Rightarrow \bar{\mathcal{A}} = -\Big(\frac{\widetilde{A}_h(\theta)}{\widetilde{B}_h(\theta)}\Big)\mathcal{A},$$

where $\widetilde{A}_h(\theta), \widetilde{B}_h(\theta)$ are the Fourier symbols of the difference operators A_h, B_h. Now

$$\mu(\theta) := \left|\frac{\widetilde{A}_h(\theta)}{\widetilde{B}_h(\theta)}\right|$$

is the amplification factor (or reduction factor) of the component θ.

High and Low frequencies:

Let Ω_h be the fine grid and Ω_{2h} (standard coarsening) be the coarse grid. Then a Fourier component on fine grid is called a high frequency if it restriction to the coarse grid is not visible there. Otherwise it is called a low frequency. First we consider one-dimensional standard coarsening:

$$\Omega_h = \{\ell_1 h : \ell_1 \in \mathbb{Z}\}, \Omega_{2h} = \{2\ell_1 h : k \in \mathbb{Z}\}.$$

Let $|\theta| \leqslant \pi$, then by injection from fine to coarse grid:

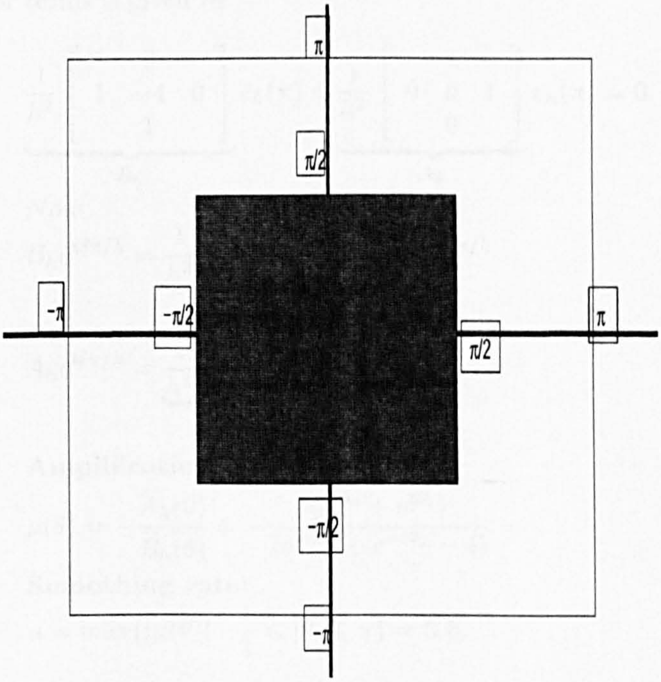
$$I_h^{2h}(e^{\mathrm{i}\theta x/h}) = e^{\mathrm{i}2\theta x/2h},$$

this component $e^{\mathrm{i}2\theta x/2h}$ is visible on coarse grid Ω_{2h} only if $|2\theta| \leqslant \pi$, i.e. $|\theta| \leqslant \frac{\pi}{2}$ the high frequencies on Ω_h are those with $\frac{\pi}{2} \leqslant |\theta| \leqslant \pi$. In 2-D, Fourier components $e^{\mathrm{i}\theta\mathbf{x}/h} = e^{\mathrm{i}\theta_1 x/h}e^{\theta_2 y/h}$, with $\theta = (\theta_1, \theta_2)$ and $|\theta| := \max(|\theta_1|, |\theta_2|) \leqslant \pi$. Then

High frequencies:	$\frac{\pi}{2} \leqslant \lvert\theta\rvert \leqslant \pi,$
Low frequencies:	$\lvert\theta\rvert < \frac{\pi}{2}.$

Definition 2.9.2 (Smoothing) *Smoothing stands for the convergence of high frequency error components which cannot be approximated from the coarse grids in a multigrid cycle.*

[9] Relaxation sweep means one step of error smoothing


π
π/2
-π
-π/2
π/2
π
-π/2
-π

Definition 2.9.3 (Smoother) *An iterative method which is used to smooth the error.*

Definition 2.9.4 (Smoothing rate) *For a relaxation scheme with amplification factors $\mu(\theta)$ of Fourier components $e^{i\theta x/h}$ the* **smoothing rate** *$\bar{\mu}$ is defined by*

$$\bar{\mu} := \max\{|\mu(\theta)| : \theta \in \textit{high frequencies}\}.$$

Smoothing rates are also known as smoothing factors.

Now to find the smoothing rate for Gauss-Seidel relaxation for Poisson's equation, consider

$$\Delta_h U_h = \frac{1}{h^2}\begin{bmatrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{bmatrix} U_h = f_h. \tag{2.90}$$

Relaxation in error terms is given by

$$\underbrace{\frac{1}{h^2}\begin{bmatrix} & 0 & \\ 1 & -4 & 0 \\ & 1 & \end{bmatrix}}_{B_h} \bar{e}_h(\mathrm{x}) + \underbrace{\frac{1}{h^2}\begin{bmatrix} & 1 & \\ 0 & 0 & 1 \\ & 0 & \end{bmatrix}}_{A_h} e_h(\mathrm{x}) = 0.$$

Now

$$B_h e^{i\theta x/h} = \underbrace{\frac{1}{h^2}(e^{-i\theta_1} + e^{-i\theta_2} - 4)}_{\widetilde{B}_h(\theta)} .e^{i\theta x/h}$$

$$A_h e^{i\theta x/h} = \underbrace{\frac{1}{h^2}(e^{i\theta_1} + e^{i\theta_2})}_{\widetilde{A}_h(\theta)} .e^{i\theta x/h}$$

Amplification factors:

$$\mu(\theta) = -\frac{\widetilde{A}_h(\theta)}{\widetilde{B}_h(\theta)} = -\frac{(e^{i\theta_1} + e^{i\theta_2})}{(e^{-i\theta_1} + e^{-i\theta_2} - 4)}$$

Smoothing rate:

$$\mu = \max\{|\mu(\theta)| : \frac{\pi}{2} \leqslant |\theta| \leqslant \pi\} = 0.5.$$

The high frequency error components are reduced by a factor of (at least) 0.5 per relaxation sweep.

Note 1 *For low frequencies, the reduction per relaxation sweep is much worse:*

$$|\mu(\theta)| \to 1 \quad if \quad \theta \to 0.$$

2.10 Restriction and Interpolation for Vertex Centered Grids

The choice of restriction and interpolation operators I_h^{2h} and I_{2h}^h, for the intergrid transfer of grid function, are closely related to the choice of the coarse grid. In this section we will introduce the restriction and interpolation operators for standard coarsening.

2.10.1 Restriction Operator

A restriction operator I_h^{2h} maps h-grid functions to $2h$-grid functions. A frequently used restriction operator is the full weighting (FW) operator, which is in stencil notation given by

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

Applying this operator to a grid function $u(x,y)$ at a coarse grid point $(x,y) \in \Omega_{2h}$ means

$$\begin{aligned} u_{2h}(x,y) &= I_h^{2h} u_h(x,y) \\ &= \frac{1}{16}[4u_h(x,y) + 2u_h(x+h,y) + u_h(x-h,y) + 2u_h(x,y+h) + 2u_h(x,y-h) \\ &+ u_h(x+h,y+h) + u_h(x+h,y-h) + u_h(x-h,y+h) + u_h(x-h,y-h)]. \end{aligned}$$

Another restriction operator is known as half weighting (HW):

$$\frac{1}{8}\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}_h^{2h} .$$

2.10.2 Interpolation

The interpolation operators (prolongation operators) map $2h$-grid functions into h-grid functions. A very frequently used interpolation operator is the bilinear interpolation from Ω_{2h} to Ω_h, which is given by (with Figure 2.11):

$$I_{2h}^h e_{2h}(x,y) = \begin{cases} e_{2h}(x,y) & \text{for } \triangleleft \\ \frac{1}{2}[e_{2h}(x,y+h) + e_{2h}(x,y-h)] & \text{for } \Box \\ \frac{1}{2}[e_{2h}(x+h,y) + e_{2h}(x-h,y)] & \text{for } \diamond \\ \frac{1}{4}[e_{2h}(x+h,y+h) + e_{2h}(x+h,y-h) & \\ +e_{2h}(x-h,y+h) + e_{2h}(x-h,y-h)] & \text{for } \circ\,. \end{cases} \tag{2.91}$$

2.10.3 Restriction and Interpolation for Cell Centered Grids

The cell-centered discretization uses unknowns located at the centers of the grid cells figure 2.12. In the case of Poisson equation, there is no difference in the order of accuracy of the solution of a vertex or a cell centered discretization. The treatment of boundary conditions is different in vertex and cell centered discretization. Efficient multigrid methods can also be developed for cell-centered discretization. The main difference in the multigrid algorithm is that the coarse grid points do not form a subset of fine grid points.

Restriction:

If we use a cell-centered discretization, each cell of the coarse grid Ω_{2h} contains within it 4 fine grid cells and each mesh point of Ω_{2h} is surrounded by the 4 mesh points of Ω_h. The four cell average restriction operator evaluates the value of a coarse grid function u_{2h} at a coarse grid point by taking the average value of the fine grid function u_h at the

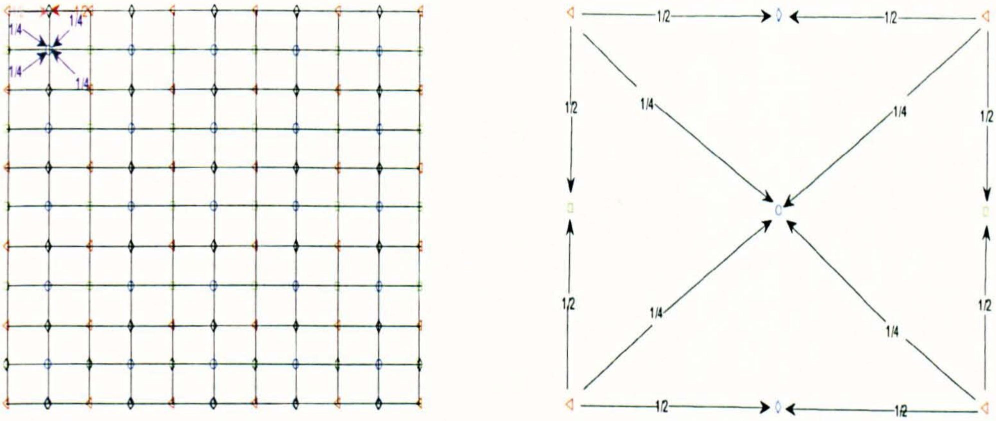


Figure 2.11: Left: A fine grid with symbols indicating the bilinear interpolation used for the transfer from the coarse grid (◁). Right: The distribution process for the bilinear interpolation operator .

four fine grid points surrounding it. The restriction operator is given by

$$\begin{aligned} u_{2h}(x,y) &= I_h^{2h} u_h(x,y) = \frac{1}{4}\begin{bmatrix} 1 & & 1 \\ & \cdot & \\ 1 & & 1 \end{bmatrix}_h^{2h} u_h(x,y) \\ &= \frac{1}{4}[u_h(x-\frac{h}{2}, y-\frac{h}{2}) + u_h(x-\frac{h}{2}, y+\frac{h}{2}) \\ &+ u_h(x+\frac{h}{2}, y-\frac{h}{2}) + u_h(x+\frac{h}{2}, y+\frac{h}{2})]. \end{aligned}$$

Interpolation:

The bilinear interpolation operator for cell centered discretization is given by:

$$e_h = I_{2h}^h e_{2h}$$

where

$$\begin{aligned} e_h^{2i,2j} &= \frac{1}{16}[9e_{2h}^{i,j} + 3(e_{2h}^{i+1,j} + e_{2h}^{i,j+1}) + e_{2h}^{i+1,j+1}], \\ e_h^{2i+1,2j} &= \frac{1}{16}[9e_{2h}^{i+1,j} + 3(e_{2h}^{i,j} + e_{2h}^{i+1,j+1}) + e_{2h}^{i,j+1}], \\ e_h^{2i,2j+1} &= \frac{1}{16}[9e_{2h}^{i,j+1} + 3(e_{2h}^{i,j} + e_{2h}^{i,j+1}) + e_{2h}^{i+1,j+1}], \\ e_h^{2i+1,2j+1} &= \frac{1}{16}[9e_{2h}^{i+1,j+1} + 3(e_{2h}^{i+1,j} + e_{2h}^{i,j+1}) + e_{2h}^{i,j}]. \end{aligned}$$

In stencil notation we have

$$\begin{bmatrix} 1 & 3 & 3 & 1 \\ 3 & 9 & 9 & 3 \\ 3 & 9 & 9 & 3 \\ 1 & 3 & 3 & 1 \end{bmatrix}_{2h}^{h}.$$

More details can be found in [39, 68, 118].

Order of Interpolation and Restriction

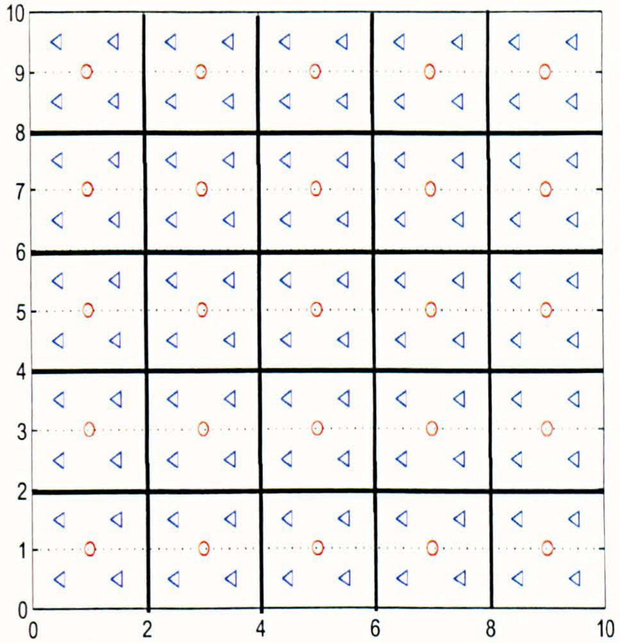


Figure 2.12: Cell-Centered discretization, ($\circ$) denotes fine grid points and ($\triangleleft$) are coarse grid points.

An interpolation operator is said to have an order $p+1$ if it can transfer exactly polynomials of order p i.e. if the exact values of a polynomial are given at the coarse grid points, the exact value of the polynomial can be found at all fine grid points by interpolating with the given operator. The order of a restriction operator is equal to the order of its transpose. Bilinear interpolation in both the vertex and cell-centered cases has order 2, which means the full weighting restriction operator also has order 2.

Remark 2.10.1 *Let p be the order of the operator L in the differential equation $Lu = f$. Let p_r and p_i are the orders of restriction and interpolation operators respectively, then the order of the transfer operators should fulfill [118]*

$$p_r + p_i > p.$$

In this thesis we mainly solve PDEs of order 2, so we use bilinear interpolation and full weighting restriction operators (both are of order 2) as they fulfill the above condition.

2.11 Coarse Grid correction

One idea to approximately solve the residual equation is to use an appropriate approximation L_{2h} of L_h on a coarse grid Ω_{2h}. The residual equation becomes

$$L_{2h}\hat{e}^n_{2h} = r^n_{2h},$$

assume that L^{-1}_{2h} exist.

Coarse Grid Correction: $u^m_h \to u^{m+1}_h$.

- Compute residual $r_h^m = f_h - L_h u_h^m$.
- Restrict the residual to coarse grid $r_{2h}^m = I_h^{2h} e_h^m$.
- Solve on coarse grid Ω_{2h} $L_{2h}\hat{e}_{2h}^m = r_{2h}^m$.
- Interpolate $\hat{e}_{2h}^m$ to fine grid $\hat{e}_h^m = I_{2h}^h \hat{e}_{2h}^m$.
- Compute the new approximation $u_h^{m+1} = u_h^m + \hat{e}_h^m$.

The operator L_{2h} is usually the direct analogue of L_h on the grid Ω_{2h} i.e. the discrete operator which results from discretizing the continuous problem Ω_{2h}. An alternative is the Galerkin approach which defines L_{2h} as $I_h^{2h} L_h I_{2h}^h$. The Galerkin approach is often combined with more sophisticated, matrix dependent interpolation operators [28, 49, 48] used for more difficult problems in which the coarse grid operator is not well approximated by re-discretization, and within the purely black box algebraic multigrid methods to automatically define an accurate coarse grid problem.

2.12 Two Grid Algorithm

Each iteration (cycle) of multigrid method consists of pre-smoothing, a coarse grid correction and post smoothing steps. The algorithm is given below

Algorithm 3 (Two Grid Algorithm) $u_h^{m+1} = TWOGRID(u_h^m, L_h, f_h, \nu_1, nu_2)$ ν_1: *No. of pre-smoothing steps.*
ν_2: *No. of post-smoothing steps.*

- *Smooth the solution on the fine grid using appropriate smoother i.e*

$$\bar{u}_h^m = SMOOTHER^{\nu_1}(L_h, u_h^m, f_h)$$

- *Compute the residual* $\bar{r}_h^m = f^h - L_h \bar{u}_h^m$.
- *Restrict* $\bar{r}_h^m$ *to coarse grid* $\bar{r}_{2h}^m = I_h^{2h} \bar{r}_h^m$.
- *Solve exactly on coarse grid* Ω_{2h}, $L_{2h}\hat{e}_{2h}^m = \bar{r}_{2h}^m$.
- *Interpolate the correction* $\hat{e}_h^m = I_{2h}^h \hat{e}_{2h}^m$.
- *Compute the new approximation* $u_h^m = \bar{u}_h^m + \hat{e}_h^m$.
- *Apply* ν_2 *steps of smoother (post-smoothing) i.e*

$$u_h^{m+1} = SMOOTHER^{\nu_2}(L_h, u_h^m, f_h)$$

2.13 Multigrid Algorithm

Recursive use of "Two Grid" method leads to a multigrid method. Consider the following grid sequence

$$\Omega_h(finest) \supset \Omega_{2h} \supset \Omega_{4h} \supset \ldots \supset \Omega_{Lh}(coarsest).$$

Solve the equations on the finest grid Ω_h iteratively, assuming that smoothing can be adequately done on all fine grids and we can afford to solve the coarsest grid equation on Ω_{Lh} directly.

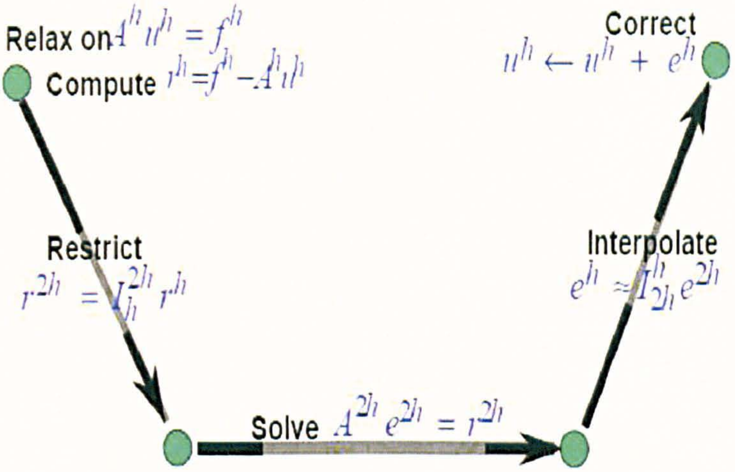

Relax on $A^h u^h = f^h$
Compute $r^h = f^h - A^h u^h$
Correct
$u^h \leftarrow u^h + e^h$
Restrict
$r^{2h} = I_h^{2h} r^h$
Interpolate
$e^h \approx I_{2h}^h e^{2h}$
Solve $A^{2h} e^{2h} = r^{2h}$

Figure 2.13: One cycle of Two Grid method [16].

Algorithm 4 (Multigrid Algorithm (MGM)) $u_k^{m+1} = MGM(\gamma, \nu_1, \nu_2, u_h^m, L_h, f_h)$*:*
To solve a discretized PDE using multigrid method we need to set the following parameters;
ν_1 *: No. of pre smoothing steps before restriction on fine grid.*
ν_2 *: No. of post smoothing steps after interpolation.*
γ *: No. of multigrid cycles (for V-cycle* $\gamma = 1$ *and for W-cycle* $\gamma = 2$*).*

- *If* Ω_h *is coarsest grid, solve* $L_h u_h = f_h$ *and STOP.*
  Else apply ν_1 *steps of smoother to relax the solution, i.e* $\bar{u}_h^m = SMOOTHER^{\nu_1}(u_h^m, L_h, f_h)$.
- *Compute residual* $\quad \bar{r}_h^m = f^h - L_h \bar{u}_h^m$.
- *Restrict* $\bar{r}_h^m$ *to coarse grid* $\quad \bar{r}_{2h}^m = I_h^{2h} \bar{r}_h^m$.
- *Compute an approximate solution* $\hat{e}_{2h}^m$ *of the residual equation on coarse grid* Ω_{2h},
$$L_{2h} \hat{e}_{2h}^m = \bar{r}_{2h}^m. \tag{2.92}$$
$$\left\{ \begin{array}{c} \textit{On the coarsest grid,} \\ \textit{solve (2.92) using any direct method.} \\ \textit{Else} \\ \bar{e}_{2h}^m = MGM(\gamma, \nu_1, \nu_2, 0, L_{2h}, \bar{r}_{2h}^m). \end{array} \right.$$
- *Interpolate the correction* $\quad \hat{e}_h^m = I_{2h}^h \bar{e}_{2h}^m$.
- *Compute the new corrected approximation on* Ω_h, $\quad u_h^m = \bar{u}_h^m + \hat{e}_h^m$.
- *Apply* ν_2 *steps of smoother (post-smoothing) i.e*
$$u_h^{m+1} = SMOOTHER^{\nu_2}(L_h, u_h^m, f_h).$$

See Figure 2.14 which shows the V-cycle of a multigrid method

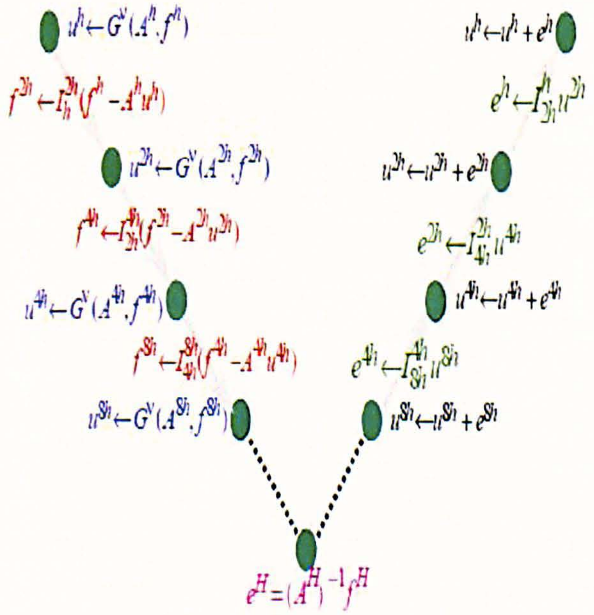

$u^h \leftarrow G^\nu(A^h, f^h)$
$u^h \leftarrow u^h + e^h$
$f^{2h} \leftarrow I_h^{2h}(f^h - A^h u^h)$
$e^h \leftarrow I_{2h}^h u^{2h}$
$u^{2h} \leftarrow G^\nu(A^{2h}, f^{2h})$
$u^{2h} \leftarrow u^{2h} + e^{2h}$
$f^{4h} \leftarrow I_{2h}^{4h}(f^{2h} - A^{2h} u^{2h})$
$e^{2h} \leftarrow I_{4h}^{2h} u^{4h}$
$u^{4h} \leftarrow G^\nu(A^{4h}, f^{4h})$
$u^{4h} \leftarrow u^{4h} + e^{4h}$
$f^{8h} \leftarrow I_{4h}^{8h}(f^{4h} - A^{4h} u^{4h})$
$e^{4h} \leftarrow I_{8h}^{4h} u^{8h}$
$u^{8h} \leftarrow G^\nu(A^{8h}, f^{8h})$
$u^{8h} \leftarrow u^{8h} + e^{8h}$
$e^H = (A^H)^{-1} f^H$

Figure 2.14: One cycle of Multi Grid method [16]

2.14 Non-Linear Multigrid

In many cases local nonlinear relaxation methods such as Gauss-Seidel Newton have a similar smoothing effect on the error to their linear analogues and the same principles of recursive application of smoothing and coarse grid correction that are used to construct linear multigrid methods can also be applied to discrete nonlinear problems [14, 39].

2.14.1 Non-Linear Residual Equation

Assume that we have a discrete nonlinear equation

$$N_h(u_h) = f_h$$

on some fine grid Ω_h, then the residual equation for non-linear discretized PDE on the fine grid Ω_h is

$$N_h(u_h) - N_h(v_h) = N_h(v_h + e_h) - N_h(v_h) = r_h,$$

where v_h is the approximation to u_h, e_h is the error in v_h and $r_h = f_h - N_h(v_h)$ is the residual. This residual equation on the coarse grid Ω_{2h} becomes

$$N_{2h}(v_{2h} + e_{2h}) - N_{2h}(v_{2h}) = r_{2h}.$$

The algorithm for nonlinear multigrid can be described in similar lines as done for the linear multigrid 2.13.

2.15 Full Multigrid Method

When convergent, multigrid iterations typically converge at a rate independent of the problem size. They will thus provide a solution with prescribed accuracy in a fixed number of iterations, independent of how fine the mesh is. However, the finer the mesh, the better accuracy can be expected, and thus the more and more iterations are needed to apply to exploit the accuracy of the discretization. In total, this effect contributes a logarithmic factor to the complexity estimate. This changes, when better initial values are used to start the multigrid iteration. The natural technique is nested iteration, where a multigrid method on the coarser level is used to supply an initial guess by interpolating its result. If this is used recursively, the so-called full multigrid method results. It starts on the coarsest discretization with an exact solver. These results are interpolated to the next finer grid, where a few cycles (V or W) of the multigrid method are applied. The result is again interpolated to the next finer grid, where again a few cycles of multigrid suffice to produce a solution whose algebraic accuracy and differential accuracy match. This algorithmic scheme typically requires just one or two V-cycles on each level to maintain truncation error accuracy on each level. The resulting method has optimal complexity in the sense that it produces solutions at a cost proportional to the number of unknowns. In Figure 2.15, full multigrid scheme is given.

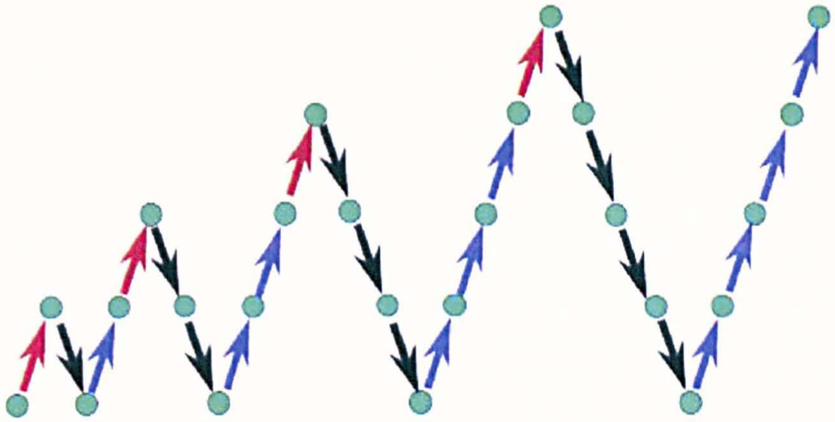


Figure 2.15: Full Multigrid scheme [16]. Blue arrow: Interpolation; Red arrow: High order interpolation; Black arrow: Restriction. On coarsest level, we solve the problem exactly and on finer level only we do only smoothing.

Chapter 3

Review of Variational Models in Image Processing

In the broadest sense, image processing is a form of information processing for which both the input and output are images, such as photographs or frames of video. Most image processing techniques involve treating the image as a two-dimensional (2-D) signal and applying standard signal processing techniques to it, where **signal processing** refers to the processing and interpretation of signals. In this chapter we mainly focus on the variational models for Image De-noising and Image Segmentation. Various existing numerical methods for the realization of these models will be discussed.

3.1 How do we define an Image?

In Digital (discrete) sense a grey image is a 2-D array of numbers (matrix), while a color image is a 2-D array of vectors (Red Green Blue, shortly RGB). In continuous sense a grey image is a 2-D function $z(x,y)$ and color image is a 2-D vector function $(r(x,y), g(x,y), b(x,y))$. In this thesis we mainly work on grey value images, which normally take values in the range $[0,255]$. An image is usually divided into $m_1 \times m_2$ parts, each part is called pixel. Each pixel value in the array represents the average light intensity over a small part (pixel) of the image. In Figure 3.1, examples of image are given which will be used for our experimental work in coming chapters. Images in first row are specially used for 2-phase image segmentation, images in second row are used for multiphase segmentation and the third row images are used for segmentation of special features in an image.

3.2 Variational Models and Partial Differential Equations

Digital images are representations of the visual world surrounding us. The common point between all digital images is the fact that they are defined in a discrete setting although they come from a continuous world. The transfer process is done by sampling and quantizing the "continuous images". Even if all image processing methods are developed for digital/discrete images, it is often more convenient to use continuous formulations. At the beginning of image processing history, the techniques used to process images such as filter theory or spectral analysis were based on a discrete setting. Nowadays, new techniques such as wavelets theory or variational models are based on a continuous setting.

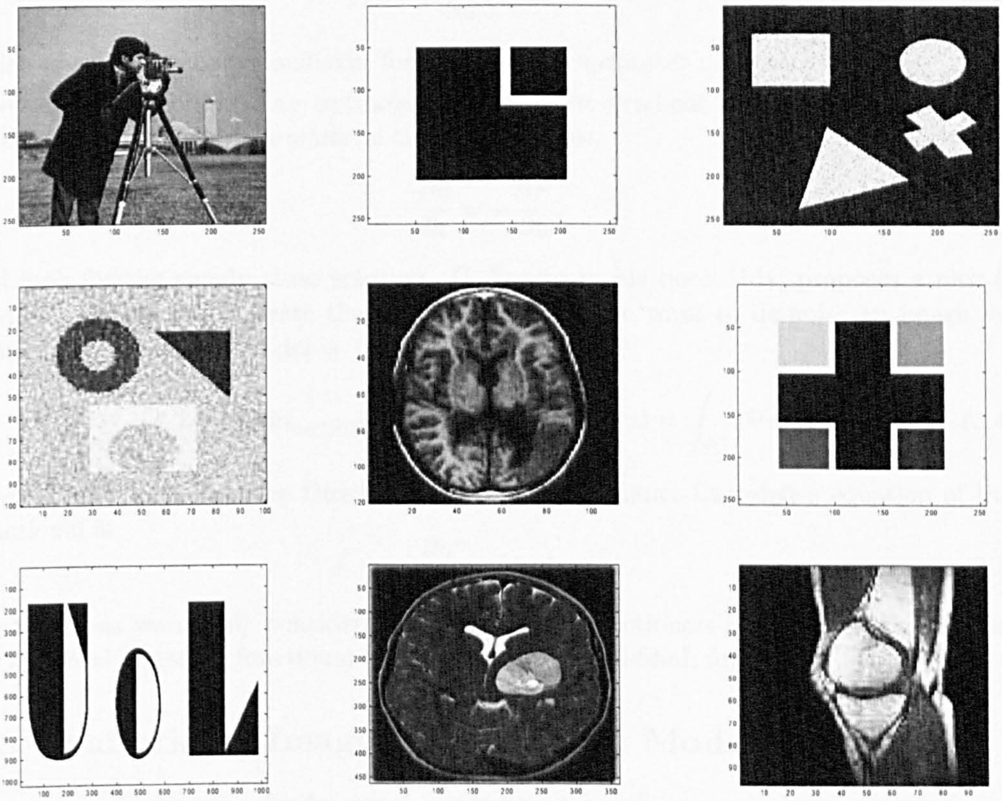


Figure 3.1: Sample images used in our experiments. Images in top row are used for 2-phase image segmentation, the images in second row are used for multiphase image segmentation and images in third row are used for segmentation under geometrical conditions.

In this thesis we develop fast numerical methods for solution of partial differential equations based models and variational models for solving image segmentation problems. Let us assume that an image processing problem can be formulated as

$$u^* = \text{argmin}_{u \in S} F(u), \tag{3.1}$$

where u^*, defined in an appropriate space S for the given problem, is an optimizer of an energy functional $F(\cdot)$ which gives solution to the given image processing problem. If F is continuous and differentiable, it is possible to compute the first variation to determine the Euler-Lagrange's equation

$$\frac{\partial F}{\partial u} = 0, \tag{3.2}$$

which gives a necessary condition for u^* to be an optimizer of F such that $\left.\frac{\delta F}{\delta u}\right|_{u^*} = 0$. Then a way to compute an optimizer is to use the gradient descent (time marching) method by introducing an artificial time t such that:

$$\frac{\partial u}{\partial t} = -\frac{\partial F}{\partial u}, \tag{3.3}$$

and look for the steady state solution. G. Sapiro in his book [104] proposes a nice (a simple) example to illustrate the previous ideas. If we want to de-noise an image, an example of variational model is

$$u^* = \text{argmin}_{u \in L^2(\mathbf{R}^2)} F(u) \qquad \text{where } F(u) = \int_{\mathbf{R}^2} |\nabla u|^2 dx, \tag{3.4}$$

where F is in this case the Dirichlet functional. The Euler-Lagrange's equation of this functional is

$$\frac{\partial u^*}{\partial t} = \Delta u^*. \tag{3.5}$$

In this thesis we mainly consider the fundamental functionals like Rudin-Osher-Fatemi (ROF) total variation functional [99] and the Mumford-Shah functional [88].

3.3 Variational Image Segmentation Models

Image segmentation is a fundamental component towards automated vision systems and is useful in medical applications. Segmentation means dividing an image into a patchwork of regions, (domains) each of which is homogeneous, that is, they are same in some sense like intensity, texture, color etc, in other words distinguishing objects from the background or systematically select specific features out of an image that has many features [33, 6, 87]. For intensity based images, the non-equation based methods are the popular approaches: threshold techniques, region merging algorithms, the watershed segmentation techniques and so on. Most of them are based on a discrete setting, which makes them dependent on the parametrization. Moreover, they are not defined in rigorous mathematical framework, for more details see [111]. One may also view the task of distinguishing objects of interest from "the rest", is to identify the feature boundaries. In recent years, a class of image segmentation models based on variational approaches have been introduced. These are defined in continuous setting and are mathematically well studied. Two well known variational image segmentation models are the Mumford-Shah model [88] and the active contour model [75]. In the Mumford-Shah approach, the goal is to find a partition of the given image z into different homogeneous regions, in terms of intensity, color, texture etc. The n-dimension Mumford-Shah functional can be defined as

$$F^{MS}(u,K)=\beta\int_{\Omega}(u-z)^2dxdy+\int_{\Omega\setminus K}|\nabla u|^2dxdy+\alpha\mathcal{H}^{n-1}(K), \tag{3.6}$$

where Ω is a bounded domain in $\mathbb{R}^n, z:\Omega\to[0,1]$ is a given function (input grey level image), α and β are positive (tuning) parameters, $\mathcal{H}^{n-1}$ is the $(n-1)$-dimensional Hausdorff measure (that is, the usual $(n-1)$-dimensional area in case of subsets of regular hyper surfaces, the length in the most relevant case $n=2$). The unknown function $u:\Omega\to\mathbb{R}$ is regular (say, of class C^1) out of a closed singular set K, whose shape and location are not prescribed. In equation (3.6), the first term is the fitting (fidelity) term with respect to the given image z, it asks that u approximates z. The second term is the regularization term which asks u to be smooth inside the $\Omega\setminus K$ and the last term is the constraint on the set of discontinuities (edges) K which asks the boundaries to be as short as possible. Theoretical results on the existence and regularity of minimizers of (3.6) can be found in [88, 87, 86] and [58]. A reduced case of the above model is obtained by assuming u as a piecewise constant function inside each connected region say Ω_i, i.e $u=c_i$ inside each connected region Ω_i. Thus Mumford and Shah model is reduced to minimizing the following functional:

$$F^{MS}_{RC}(u,K)=\beta\sum_i\int_{\Omega_i}(z-c_i)^2dxdy+\alpha|K|. \tag{3.7}$$

As we mainly consider problems in 2-dimension, the last term of (3.6) is the total length of the set of discontinuities K. For fixed K, minimizing (3.7) with respect to c_i gives

$$c_i=\frac{\int_{\Omega_i}zdxdy}{\int_{\Omega_i}dxdy}\equiv\text{mean}(z)\qquad in\quad\Omega_i.$$

Hence (3.7) reduces to the following minimization problem

$$F^{MS}_{RC}(K)=\beta\sum_i\int_{\Omega_i}(z-\text{mean}(z))^2dxdy+\alpha|K|. \tag{3.8}$$

Theoretical results for the existence and regularity of minimizers of (3.7) can be found for example in [88], [116] and references there in. In practice the functionals (3.6) and (3.7) are difficult to minimize because of the unknown set K, is of lower dimension and also the functionals are not convex. A possible solution to these problems will be addressed in Section 3.5 and in chapter 5.

Although the first segmentation model (3.6) proposed is to extract all significant parts in images, some specific parts of image can be more important than others depending on applications such as in medical imaging (look for tumor in a brain MRI). This makes the link with the second segmentation model, in the context of variational models, which aims at detecting edges in images. The active contour model proposes to detect the closest contour(s) from an initial position. The active contour/snake was initially introduced by Kass et. al in [75]. The active contour model locates sharp image intensity variations by deforming a curve C towards the edges of objects in a given image z. The snake model has external constraint forces and the internal constraint forces in its energy functional. The image forces push the contour/snake toward image features like lines, edges etc, the internal forces impose the smoothness constraint and the external constraint forces put the contour/snake near the desired local minimum. The evolution equation (introducing artificial time step) of the parametric planar curve $C(q) = (x(q), y(q)) \in \Omega, q \in [0,1]$ is given by the minimization of the following energy functional:

$$F_{KWT}(C(q)) = \alpha \int_0^1 |\frac{\partial C(q)}{\partial q}|^2 dq + \beta \int_0^1 |\frac{\partial^2 C(q)}{\partial^2 q}|^2 dq + \lambda \int_0^1 g^2(\nabla(z * G_\sigma)(c(q)))dq, \quad (3.9)$$

where α, β and λ are positive constants. Also $g : [0, +\infty[\rightarrow \mathbb{R}^+$ is called the edge detecting function, and is a strictly decreasing function which vanishes at $+\infty$ and is given by

$$g(\nabla(z * G_\sigma)) = \frac{1}{1 + \gamma|\nabla(z * G_\sigma)|^2}, \quad (3.10)$$

where $G_\sigma(x,y) = \frac{1}{2\pi\sigma^2} \exp\left((x - \mu_x)^2 + (y - \mu_y)^2/2\sigma^2\right)$ is the Gaussian function with standard deviation σ and mean μ_x, μ_y, $z * G_\sigma$ is a smooth version of the original image z and γ is a positive constant. F_{KWT} is not convex [75], so the solution will not be unique and it has local minima. So the local minima of F_{KWT} can be reached by solving the following Euler-Lagrange's equation

$$-\alpha \frac{\partial^2 C}{\partial q^2} + \beta \frac{\partial^4 C}{\partial q^4} + \lambda \nabla g^2 = 0. \quad (3.11)$$

In [75], this fourth order equation (3.11) was numerically solved using finite differences. It provides a fast numerical algorithm but, in the case of closed curves, it does not allow changes of topology since the final curve has the same topology as the initial one. In other words, it is not possible to detect more than one object. Another drawback of the snake segmentation model is the dependence of the functional with respect to the parametrization of the curve C, i.e different parametrization of the curve may give different solutions for the same initial condition.

To overcome the limitation of the changes of topology, the powerful level set method [90, 108, 92] may be used. The curve C is then implicitly represented by a function of higher dimension ϕ, called the level set function, and the curve evolution equation can be re-written in a level set formulation.

3.4 The Geodesic Active Contours Model

V. Casselles et al [21] proposed a new and improved energy, based on Kass et al [75] that is invariant with respect to a new curve parametrization [21, 103, 60, 100, 96]. The new energy is given by

$$F_{GAC}(C(q)) = \int_0^1 g(|\nabla z(C(q))|)|C'(q)|dq. \quad (3.12)$$

Since $L(C) = \int_0^1 |C'(q)|dq = \int_0^{L(C)} ds$, where ds is the Euclidean element of length and $L(C)$ is the Euclidean length of the curve C, hence equation (3.12) becomes

$$F_{GAC}(C(q)) = \int_0^{L(C)} g(|\nabla z(C(q))|)ds \quad (3.13)$$

Here the functional (3.13) is actually a new length obtained by weighting the Euclidean element of length ds by the function g which contains information concerning the boundaries of objects [7]. The function g is the edge detecting function defined in (3.10). The equivalence between minimizing F_{GAC} and minimizing F_{KWT} at $\beta = 0$ was studied in [21]. Hence the direction for which F_{GAC} decreases most rapidly provides us the following minimization flow: more details of its derivation can be found in [21]:

$$\frac{\partial C}{\partial t} = g\kappa\vec{\mathcal{N}} - (\nabla g \cdot \vec{\mathcal{N}})\vec{\mathcal{N}}, \tag{3.14}$$

where κ is the Euclidean curvature, and $\mathcal{N}$ is the unit normal vector. This equation shows how each point in the active contour C should move in order to decrease the length F_{GAC}. The detected object is then given by the steady state solution of (3.14). In level set formulation, the evolution equation is:

$$\frac{\partial \phi}{\partial t} = |\nabla\phi|(\nabla \cdot (g\frac{\nabla\phi}{|\nabla\phi|}) + \nu_1 g), \tag{3.15}$$

ϕ is a Lipschitz function representing C as a zero level set. The term $\nu_1 g$, $\nu_1 > 0$ is added to increase the evolution speed and to attract the curve towards the boundary and constitute in fact an extra area-based speed.

These snakes/active contour models rely on the edge function g which depends on the gradient of the image. These models can detect only objects with edges defined by gradient. Also, in practice the discrete gradients are bounded and then the stopping function g is never zero at the edges, and the contour may pass through the image edges [35]. On the other hand if the given image is too noisy, then the smoothing Gaussian has to be strong (potentially distorted by noise), and which will smooth the edges too.

In the next section we describe another active model which does not depend on the edge function to stop the contour at edges. This is the Chan-Vese model [35], "Active Contour without Edges".

3.5 Active Contour without Edges

T. F Chan and L. A. Vese in [35] proposed a new energy based model for image segmentation, which does not use the gradient of image z as a stopping process, but the stopping term is depending on Mumford-Shah segmentation technique [88]. This means that this model can detect contours with or without gradients. The basic idea of contour models or snakes is to evolve a curve, subject to constraints from a given image in order to detect objects in that image. Let z be the given image, as a bounded function. Assume that z is formed by two regions of approximately constant intensities of distinct values z_i and z_o. Assume that the object to be detected is represented by the region with intensity z_i and its boundary is Γ_0. Consider the 'fitting' term

$$E_1(\Gamma) = \int_{inside(\Gamma)} |z - c_1|^2 dxdy + \int_{outside(\Gamma)} |z - c_2|^2 dxdy, \tag{3.16}$$

where Γ is the unknown evolving curve and c_1 and c_2 are the averages of the image z inside and outside of Γ respectively. In this case, it is obvious that Γ_0, the boundary of the object, is the minimizer of the energy. To minimize (3.16), some regularization is required. The regularization terms used in [35] are the length of Γ and the area of the region inside Γ. Thus similar to (3.7) they consider the following energy:

$$\begin{aligned} F(\Gamma, c_1, c_2) &= \mu.(length(\Gamma))^p + \nu.area(inside(\Gamma)) + \\ \lambda_1 \int_{inside(\Gamma)} |z - c_1|^2 dxdy &+ \lambda_2 \int_{outside(\Gamma)} |z - c_2|^2 dxdy, \end{aligned} \tag{3.17}$$

where c_1 and c_2 are unknown constants, $\mu \geqslant 0$, $\nu \geqslant 0$, $\lambda_1, \lambda_2 > 0$ are fixed parameters. In [35] $\lambda_1 = \lambda_2 = 1$ and $p > 0$, in particular we take $p = 1$. Γ generally mean a hypersurface in $\mathbb{R}^n$ and "$length(\Gamma)$" means the Hausdorff $(n-1)$-dimensional measure $\mathcal{H}^{n-1}(\Gamma)$. If the curve is forced to move only inside or outside then we take positive ν otherwise $\nu = 0$. We ignore the area term in later chapters i.e. $\nu = 0$.

Thus Chan and Vese in [35] proposed the following minimization problem

$$\inf_{\Gamma, c_1, c_2} F(\Gamma, c_1, c_2). \tag{3.18}$$

This functional is a special case of the piecewise constant Mumford and Shah segmentation model [88] (restricted to only two regions with constant average intensities).

Level Set formulation of the Model:

Let $\phi : \mathbb{R}^2 \to \mathbb{R}$ be a Lipschitz function [92], then representing Γ as a zero level set of ϕ, such that

$$\begin{cases} \Gamma = \{(x,y) \in \mathbb{R}^2 : \phi(x,y) = 0\}, \\ inside(\Gamma) = \{(x,y) \in \mathbb{R}^2 : \phi(x,y) > 0\}, \\ outside(\Gamma) = \{(x,y) \in \mathbb{R}^2 : \phi(x,y) < 0\}. \end{cases}$$

Thus the unknown lower dimensional variable curve Γ is replaced by another unknown higher dimensional variable ϕ. Let us define the Heaviside function H and the 1-dimensional Dirac measure δ concentrated at 0, defined respectively by

$$H(x) = \begin{cases} 1 & \text{if } x \geqslant 0 \\ 0 & \text{if } x < 0 \end{cases}$$

and

$$\delta(x) = \frac{d}{dx} H(x).$$

Expressing each term of the energy F in terms ϕ:

$$\begin{aligned} length\{\phi = 0\} &= \int_\Omega |\nabla H(\phi)| dxdy = \int_\Omega \delta(\phi) |\nabla \phi| dxdy, \\ area\{\phi \geqslant 0\} &= \int_\Omega H(\phi) dxdy, \\ \int_{\phi \geqslant 0} |z - c_1|^2 dxdy &= \int_\Omega |z - c_1|^2 H(\phi) dxdy, \\ \int_{\phi < 0} |z - c_2|^2 dxdy &= \int_\Omega |z - c_2|^2 (1 - H(\phi)) dxdy. \end{aligned}$$

In level set formulation equation (3.17) becomes:

$$\begin{aligned} F(\phi, c_1, c_2) = \mu \int_\Omega |\nabla H(\phi)| dxdy + \nu \int_\Omega H(\phi) dxdy + \\ \lambda_1 \int_\Omega |z - c_1|^2 H(\phi) dxdy + \lambda_2 \int_\Omega |z - c_2|^2 (1 - H(\phi)) dxdy. \end{aligned} \tag{3.19}$$

Once ϕ is obtained then the segmented image is given by

$$u = c_1 H(\phi) + c_2 (1 - H(\phi)).$$

Existence of minimizers is discussed in detail in Chan-Vese paper [35]. To minimize (3.19), we decouple the variables. Firstly keeping ϕ fixed and minimizing (3.19) with respect to c_1 and c_2 we have

$$c_1(\phi) = \frac{\int_\Omega zH(\phi)dxdy}{\int_\Omega H(\phi)dxdy} \tag{3.20}$$

if $\int_\Omega H(\phi)dxdy > 0$ (i.e the curve has a nonempty interior in Ω) otherwise need to re-initialize ϕ, and

$$c_2(\phi) = \frac{\int_\Omega z(1-H(\phi))dxdy}{\int_\Omega(1-H(\phi))dxdy} \tag{3.21}$$

if $\int_\Omega(1-H(\phi))dxdy > 0$ (i.e the curve has a nonempty exterior in Ω) otherwise re-initialize ϕ.

Since H is not differentiable at 0, to find the Euler-Lagrange equation for the unknown function ϕ we need a regularized H and δ. Let us denote the regularized version of Heaviside H and delta δ functions denoted by H_ϵ and δ_ϵ respectively and defined as in [35] (see also [39, 11]):

$$\begin{aligned} H_\epsilon(x) &= \frac{1}{2}(1+\frac{2}{\pi}\arctan(\frac{x}{\epsilon})), \\ \delta_\epsilon(x) &= H'_\epsilon(x) = \frac{1}{\pi}.\left(\frac{\epsilon}{\epsilon^2+x^2}\right). \end{aligned} \tag{3.22}$$

Thus the regularized functional denoted by F_ϵ will be:

$$\begin{gathered} F_\epsilon(\phi,c_1,c_2) = \mu\int_\Omega|\nabla H_\epsilon(\phi)|dxdy + \nu\int_\Omega H_\epsilon(\phi)dxdy + \\ \lambda_1\int_\Omega|z-c_1|^2H_\epsilon(\phi)dxdy + \lambda_2\int_\Omega|z-c_2|^2(1-H_\epsilon(\phi))dxdy. \end{gathered} \tag{3.23}$$

And the minimization problem is

$$\inf_{\phi,c_1,c_2} F_\epsilon(\phi,c_1,c_2).$$

To compute the Euler-Lagrange equation, let us keep c_1 and c_2 fixed, and minimize F_ϵ with respect to the unknown ϕ. We proceed in the following way. Let us choose ψ as a test function of the same type as ϕ and find the Gâteaux derivative of F_ϵ:

$$\begin{gathered} \lim_{t_1\to 0}\frac{1}{t_1}\Big(F_\epsilon(\phi+t_1\psi,c_1,c_2) - F_\epsilon(\phi,c_1,c_2)\Big) = 0 \\ \Rightarrow \int_\Omega \mu\Big(\delta'_\epsilon(\phi)|\nabla\phi|\psi + \delta_\epsilon(\phi)\frac{\nabla\phi\cdot\nabla\psi}{|\nabla\phi|}\Big)dxdy + \\ \int_\Omega \delta_\epsilon(\phi)(\nu+\lambda_1(z-c_1)^2 - \lambda_2(z-c_2)^2)\psi dxdy = 0 \end{gathered} \tag{3.24}$$

where

$$\int_\Omega|\nabla H_\epsilon(\phi)|dxdy = \int_\Omega \delta_\epsilon(\phi)|\nabla\phi|.dxdy$$

From Green's theorem we have

$$\int_\Omega v\nabla\cdot\vec{w}dx = -\int_\Omega \nabla v\cdot\vec{w}dx + \int_{\partial\Omega} v\vec{w}\cdot\vec{n}ds.$$

$$Let\ \psi = v \qquad and \qquad \frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\nabla\phi = \vec{w}.$$

So

$$\int_\Omega \psi\nabla\cdot\Big(\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\nabla\phi\Big)dxdy = -\int_\Omega \nabla\psi\cdot\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\nabla\phi dxdy + \int_{\partial\Omega}\psi\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\nabla\phi\cdot\vec{n}ds$$

which implies that

$$\int_\Omega \delta_\epsilon(\phi)\frac{\nabla\phi\cdot\nabla\psi}{|\nabla\phi|}dxdy = -\int_\Omega \psi\nabla\cdot\Big(\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\nabla\phi\Big)dxdy + \int_{\partial\Omega}\psi\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\frac{\partial\phi}{\partial n}ds \tag{3.25}$$

where $\nabla\phi\cdot\vec{n} = \dfrac{\partial\phi}{\partial n}$. Thus (3.24) becomes

$$\mu\int_\Omega \delta'_\epsilon(\phi)|\nabla\phi|\psi - \mu\int_\Omega \psi\nabla\cdot\Big(\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\nabla\phi\Big)dxdy + \mu\int_{\partial\Omega}\psi\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\frac{\partial\phi}{\partial n}ds +$$
$$\int_\Omega \delta_\epsilon(\phi)(\nu + \lambda_1(z-c_1)^2 - \lambda_2(z-c_2)^2)\psi dxdy = 0$$

which implies that

$$\mu\int_\Omega \delta'_\epsilon(\phi)|\nabla\phi|\psi dxdy - \mu\int_\Omega \delta_\epsilon(\phi)\nabla\cdot\Big(\frac{\nabla\phi}{|\nabla\phi|}\Big)\psi dxdy - \mu\int_\Omega \delta'_\epsilon(\phi)\nabla\phi.\frac{\nabla\phi}{|\nabla\phi|}\psi dxdy +$$
$$\mu\int_{\partial\Omega}\psi\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\frac{\partial\phi}{\partial n}ds + \int_\Omega \delta_\epsilon(\phi)(\nu + \lambda_1(z-c_1)^2 - \lambda_2(z-c_2)^2)\psi dxdy = 0.$$

Finally we obtain

$$-\int_\Omega \mu\delta_\epsilon(\phi)\nabla\cdot\Big(\frac{\nabla\phi}{|\nabla\phi|}\Big)\psi dxdy + \int_{\partial\Omega}\mu\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\frac{\partial\phi}{\partial n}\psi ds +$$
$$\int_\Omega \delta_\epsilon(\phi)(\nu + \lambda_1(z-c_1)^2 - \lambda_2(z-c_2)^2)\psi dxdy = 0.$$

for all test function ψ. Choosing $\psi \in C^1_c(\Omega)$ which is arbitrary, we deduce the following Euler-Lagrange's equation for ϕ:

$$\begin{cases} \delta_\epsilon(\phi)\Big[\mu\nabla\cdot\Big(\frac{\nabla\phi}{|\nabla\phi|}\Big) - \nu - \lambda_1(z-c_1)^2 + \lambda_2(z-c_2)^2\Big] = 0 & \text{in } \Omega, \\ \dfrac{\delta_\epsilon(\phi)}{|\nabla\phi|}\dfrac{\partial\phi}{\partial n} = 0 \quad \text{or} \quad \dfrac{\partial\phi}{\partial n} = 0 & \text{on } \partial\Omega. \end{cases} \tag{3.26}$$

In [35], the authors considered the following evolution problem, i.e the descent direction is parameterized by an artificial time t, and solve it to find the steady state solution of the following parabolic equation:

$$\begin{cases} \frac{\partial\phi}{\partial t} = \delta_\epsilon(\phi)\Big[\mu\nabla.\Big(\frac{\nabla\phi}{|\nabla\phi|}\Big) - \nu - \lambda_1(z-c_1)^2 + \lambda_2(z-c_2)^2\Big] & \text{in } \Omega, \\ \phi(t,x,y) = \phi_0(x,y) \qquad \text{in } \Omega & \\ \dfrac{\partial\phi}{\partial n} = 0 \qquad \text{on } \partial\Omega. & \end{cases} \tag{3.27}$$

To extend the evolution to all level sets of ϕ, $\delta_\epsilon(\phi)$ can be replaced by $|\nabla\phi|$ in equation (3.27) as done in [131]. This re-scaling does not affect the steady state solution, but it does remove the stiffness of the zero level sets of ϕ. But we will use $\delta_\epsilon(\phi)$ as in [35]. In next section we derive the numerical approximation of the model and will discuss the existing methods used for solving the above evolution problem (3.27).

Numerical Methods:

The following evolution problem will be solved numerically,

$$\begin{cases} c_1(\phi) = \dfrac{\int_\Omega zH_\epsilon(\phi)dxdy}{\int_\Omega H_\epsilon(\phi)dxdy} \qquad c_2(\phi) = \dfrac{\int_\Omega z(1-H_\epsilon(\phi))dxdy}{\int_\Omega(1-H_\epsilon(\phi))dxdy} \\ \dfrac{\partial\phi}{\partial t} = \delta_\epsilon(\phi)\Big[\mu\nabla.\Big(\dfrac{\nabla\phi}{|\nabla\phi|}\Big) - \nu{-}\lambda_1(z-c_1)^2 + \lambda_2(z-c_2)^2\Big] \qquad \text{in } \Omega, \\ \phi(0,x,y) = \phi_0(x,y) \qquad \text{in } \Omega, \\ \dfrac{\delta_\epsilon(\phi)}{|\nabla\phi|}\dfrac{\partial\phi}{\partial n} = 0 \qquad \text{on } \partial\Omega. \end{cases} \tag{3.28}$$

Equation (3.28) has 3 unknowns $c_1(\phi)$, $c_2(\phi)$ and ϕ, so we first compute $c_1(\phi)$ and $c_2(\phi)$ and then keep them fixed and then we solve the PDE for ϕ. Once ϕ is found then update $c_1(\phi)$ and $c_2(\phi)$ and so on. To solve the above PDE in equation (3.28), we proceed in the following way:

Let us suppose that the size of z is $m_1 \times m_2$. Finite differences scheme is used for discretization. Let $x, y \in \Omega$ be the spatial variables, h_1, h_2 be the horizontal and vertical space step size and Δt be the time step. Let $(x_i, y_j) = (ih_1, jh_2)$, for $i = 1, \dots, m_1$ and $j = 1, \dots, m_2$ be the grid points. Also let $\phi_{i,j}^k = \phi(k\Delta t, x_i, y_j)$ be an approximation of $\phi(t, x, y)$, where $k \geqslant 0$ and $\phi^0 = \phi_0$ will be given (initial guess). The finite differences are denoted by

$$\begin{gathered} \Delta_-^x\phi_{ij} = \phi_{ij} - \phi_{i-1,j}, \ \Delta_+^x\phi_{ij} = \phi_{i+1,j} - \phi_{ij}, \\ \Delta_-^y\phi_{ij} = \phi_{ij} - \phi_{i,j-1}, \ \Delta_+^y\phi_{ij} = \phi_{i,j+1} - \phi_{ij}. \end{gathered} \tag{3.29}$$

For a given ϕ^k, first compute $c_1(\phi^k)$ and $c_2(\phi^k)$ and then discretize the above parabolic PDE (3.28) and using semi implicit (SI) method to update ϕ^k. Thus discretization leads to the following equation in ϕ

$$\begin{aligned} \frac{\phi_{ij}^{k+1} - \phi_{ij}^k}{\Delta t} &= \delta_\epsilon(\phi_{ij}^k)\Big[\frac{\mu}{h_1^2}\Delta_-^x\Big(\frac{\Delta_+^x\phi_{ij}^{k+1}}{\sqrt{(\Delta_+^x\phi_{ij}^k/h_1)^2 + ((\phi_{i,j+1}^k - \phi_{i,j-1}^k)/2h_2)^2}}\Big) \\ &+ \frac{\mu}{h_2^2}\Delta_-^y\Big(\frac{\Delta_+^y\phi_{ij}^{k+1}}{\sqrt{((\phi_{i+1,j}^k - \phi_{i-1,j}^k)/2h_1)^2 + (\Delta_+^y\phi_{ij}^k/h_2)^2}}\Big) \\ &- \nu - \lambda_1(z_{ij} - c_1(\phi^k))^2 + \lambda_2(z_{ij} - c_2(\phi^k))^2\Big]. \end{aligned}$$

In this thesis we use $h_1 = h_2 = h = 1$ in all of our experiments.

$$\begin{aligned} \frac{\phi_{ij}^{k+1} - \phi_{ij}^k}{\Delta t} &= \delta_\epsilon(\phi_{ij}^k)\Big[\mu\Big(\frac{\Delta_+^x\phi_{ij}^{k+1}}{\sqrt{(\Delta_+^x\phi_{ij}^k)^2 + ((\phi_{i,j+1}^k - \phi_{i,j-1}^k)/2)^2}} \\ &- \frac{\Delta_+^x\phi_{i-1,j}^{k+1}}{\sqrt{(\Delta_+^x\phi_{i-1,j}^k)^2 + ((\phi_{i-1,j+1}^k - \phi_{i-1,j-1}^k)/2)^2}}\Big) \\ &+ \mu\Big(\frac{\Delta_+^y\phi_{ij}^{k+1}}{\sqrt{((\phi_{i+1,j}^k - \phi_{i-1,j}^k)/2)^2 + (\Delta_+^y\phi_{ij}^k)^2}} \\ &- \frac{\Delta_+^y\phi_{i,j-1}^{k+1}}{\sqrt{((\phi_{i+1,j-1}^k - \phi_{i-1,j-1}^k)/2)^2 + (\Delta_+^y\phi_{i,j-1}^k)^2}}\Big) \\ &- \nu - \lambda_1(z_{ij} - c_1(\phi^k))^2 + \lambda_2(z_{ij} - c_2(\phi^k))^2\Big]. \end{aligned}$$

$$\begin{aligned}
\Rightarrow \phi_{ij}^{k+1} &= \phi_{ij}^{k} + \Delta t \delta_\epsilon(\phi_{ij}^{k})\Big[\mu\Big(\frac{\phi_{i+1,j}^{k+1} - \phi_{i,j}^{k+1}}{\sqrt{(\Delta_+^x \phi_{ij}^k)^2 + ((\phi_{i,j+1}^k - \phi_{i,j-1}^k)/2)^2}} \\
&- \frac{\phi_{i,j}^{k+1} - \phi_{i-1,j}^{k+1}}{\sqrt{(\Delta_+^x \phi_{i-1,j}^k)^2 + ((\phi_{i-1,j+1}^k - \phi_{i-1,j-1}^k)/2)^2}}\Big) \\
&+ \mu\Big(\frac{\phi_{i+1,j}^{k+1} - \phi_{ij}^{k+1}}{\sqrt{((\phi_{i+1,j}^k - \phi_{i-1,j}^k)/2)^2 + (\Delta_+^y \phi_{ij}^k)^2}} \\
&- \frac{\phi_{ij}^{k+1} - \phi_{i,j-1}^{k+1}}{\sqrt{((\phi_{i+1,j-1}^k - \phi_{i-1,j-1}^k)/2)^2 + (\Delta_+^y \phi_{i,j-1}^k)^2}}\Big) \\
&- \nu - \lambda_1(z_{ij} - c_1(\phi^k))^2 + \lambda_2(z_{ij} - c_2(\phi^k))^2\Big].
\end{aligned}$$

Let us denote the coefficients of $\phi_{i+1,j}^{k+1}$, $\phi_{i-1,j}^{k+1}$, $\phi_{i,j+1}^{k+1}$, $\phi_{i,j-1}^{k+1}$ by A_1, A_2, A_3, A_4 respectively, we get the following system of linear equations

$$\begin{aligned}
&\phi_{ij}^{k+1}\Big[1 + \mu\delta_\epsilon(\phi_{ij}^k)(A_1 + A_2 + A_3 + A_4)\Big] \\
&= \phi_{ij}^k + \Delta t \delta_\epsilon(\phi_{ij}^k)\Big[\mu(A_1\phi_{i+1,j}^{k+1} + A_2\phi_{i-1,j}^{k+1} + A_3\phi_{i,j+1}^{k+1} + A_4\phi_{i,j-1}^{k+1}) \qquad (3.30) \\
&- \nu - \lambda_1(z_{ij} - c_1(\phi^k))^2 + \lambda_2(z_{ij} - c_2(\phi^k))^2\Big].
\end{aligned}$$

This linear system of equation can be solved by using any iterative method. To prevent the level set function to become too flat and to make it a distance function, after some steps as discussed in Section 2.4.2, it is suggested to re-initialize the function ϕ by solving the following equation [114]:

$$\begin{cases} \frac{\partial \xi}{\partial t} = \operatorname{sgn}(\phi(t))(1 - |\nabla \xi|) \\ \xi(0, t) = \phi(t), \end{cases} \qquad (3.31)$$

where $\phi(t, .)$ is the solution ϕ at time t, see [35] for more details.

Algorithm 5 (Chan-Vese (CV) algorithm for 2-phase image segmentation)

$$\phi^{k+1} \rightarrow CV(\phi^k, \mu, tol)$$

1. *For given ϕ_0, compute c_1 and c_2 using equation (3.28).*
2. *Keep c_1 and c_2 fixed and solve the PDE in equation (3.28), to update ϕ^k.*
3. *Compute c_1 and c_2 using ϕ^{k+1}.*
4. *If $|\phi^{k+1} - \phi^k| < tol$ stop else.*
5. *Re-initialize ϕ, by solving equation (3.31) and do step 2.*

This algorithm was tested on different images, as found in [35]. As mentioned before, the non linear PDE (3.28) was solved using semi implicit (SI) method, which is unconditionally stable [126]. But for higher dimensions ($n \geqslant 2$) there appears a problem: it is not possible to order the pixels in such a way that in the i-th row all non-vanishing elements of the system matrix can be found within the positions $[i, i - m_1]$ to $[i, i + m_2]$. Usually, the matrix reveals a much larger bandwidth. Applying direct algorithms would destroy the zeros within the band and would lead to an immense storage and computation effort. On the other hand, typical iterative algorithms such as the Jacobi, Gauss-Seidel, etc methods have another limitation. For large time step, the system matrix has large condition number, which causes slow convergence. J. Weickert [81, 127] proposed an Additive Operator Splitting (AOS) method for non-linear diffusion filters discussed in Section 2.7.6 and then M. Jeon [72] extended this idea toward Chan-Vese model for image segmentation. Let us consider the PDE (3.28),

$$\frac{\partial\phi}{\partial t} = \delta_\epsilon(\phi)\left[\mu\nabla\cdot(\frac{\nabla\phi}{|\nabla\phi|}) - \nu - \lambda_1(z-c_1)^2 + \lambda_2(z-c_2)^2\right]. \tag{3.32}$$

The AOS method splits the n-dimensional spatial operator into a sum of n one-dimensional space operators. As a result we get a tridiagonal system matrix. The Thomas algorithm [47] can be used to solve this tridiagonal system, resulting in a very fast and parallelizable algorithm. Let k and i represent time and spatial indices, respectively. Let $\phi_i^k = \phi(i,k)$, then at a grid the 1-dimensional semi implicit discretization of (3.32) with spatial step $h = 1$ is:

$$\frac{\phi_i^{k+1} - \phi_i^k}{\Delta t} = \delta_\epsilon(\phi_i^k)\left(\frac{\phi_{i+1}^{k+1} - \phi_i^{k+1}}{|\Delta_+^x\phi_i^k|} - \frac{\phi_i^{k+1} - \phi_{i-1}^{k+1}}{|\Delta_+^x\phi_{i-1}^k|} + F_i\right), \tag{3.33}$$

where $F_i = [-\nu - \lambda_1(z_i - c_1)^2 + \lambda(z_i - c_2)^2]$. Let

$$A_1 = \frac{1}{|\Delta_+^x\phi_i^k|} \text{ and } A_2 = \frac{1}{|\Delta_+^x\phi_{i-1}^k|},$$

so equation (3.33) becomes

$$\phi_i^{k+1} = \phi_i^k + \Delta t\delta_\epsilon(\phi_i^k)(A_1\phi_{i+1}^{k+1} - (A_1 + A_2)\phi_i^{k+1} + A_2\phi_{i-1}^{k+1} + F_i). \tag{3.34}$$

Thus with AOS method, solve problems in x-and y- directions with double time step to get two separate solutions say ϕ_1 and ϕ_2 and then find the average as

$$\phi = \frac{1}{2}(\phi_1 + \phi_2).$$

Although no stability constraint on the time-step is present when the AOS scheme is utilized, the size of the time-step cannot be very large because, splitting related artifacts associated with loss of rotational invariance will emerge. The practical implication of this is that the number of iterations needed for the contour to converge remains quite large. For images of large sizes, the methods discussed in this chapter are very slow in convergence. To avoid this problem, Multigrid method is the best option. In the next chapter we present a multigrid method for CV model [8].

3.6 Piecewise Linear and Smooth Segmentation

In this section we give the extension of two phase piecewise constant CV model to piecewise linear and smooth segmentation [119]. The piecewise linear segmentation is given by

$$\begin{aligned} &\min_{a^+,b^+,c^+,a^-,b^-,c^-,\phi} \int_\Omega \left|z(x,y) - (a^+x + b^+y + c^+)\right|^2 H(\phi(x,y))dxdy \qquad (3.35)\\ + &\int_\Omega \left|z(x,y) - (a^-x + b^-y + c^-)\right|^2 (1 - H(\phi(x,y)))dxdy + \mu\int_\Omega \left|\nabla H(\phi(x,y))\right|dxdy\\ + &\eta\int_\Omega \left((a^+)^2 + (b^+)^2\right)H(\phi(x,y))dxdy + \eta\int_\Omega \left((a^-)^2 + (b^-)^2\right)(1 - H(\phi(x,y)))dxdy \end{aligned}$$

where $a^+, b^+, c^+, a^-, b^-, c^-$ are unknown (as coefficients of polynomials of degree 1) and μ, η are positive parameters. Once ϕ is obtained then the segmented image is given by

$$u(x,y) = (a^+x + b^+y + c^+)H(\phi(x,y)) + (a^-x + b^-y + c^-)(1 - H(\phi(x,y))). \tag{3.36}$$

Minimization w.r.t $a^+, b^+, c^+, a^-, b^-, c^-, \phi$ leads to the following linear algebraic system of equations and a PDE for ϕ. The algebraic system of equations for a^+, b^+, c^+ is:

$$\begin{aligned}
a^+ \int_\Omega (x^2 + \eta)H(\phi(x,y))dxdy + b^+ \int_\Omega xyH(\phi(x,y))dxdy + c^+ \int_\Omega xH(\phi(x,y))dxdy \\
= \int_\Omega xz(x,y)H(\phi(x,y))dxdy, \\
a^+ \int_\Omega xyH(\phi(x,y))dxdy + b^+ \int_\Omega (y^2 + \eta)H(\phi(x,y))dxdy + c^+ \int_\Omega yH(\phi(x,y))dxdy \\
= \int_\Omega yz(x,y)H(\phi(x,y))dxdy, \\
a^+ \int_\Omega xH(\phi(x,y))dxdy + b^+ \int_\Omega y(\phi(x,y))dxdy + c^+ \int_\Omega H(\phi(x,y))dxdy \\
= \int_\Omega z(x,y)H(\phi(x,y))dxdy.
\end{aligned}$$

A similar system of equations can be found for a^-, b^-, c^- by replacing $H(\phi(x,y))$ by $(1 - H(\phi(x,y)))$, and the evolution problem for updating ϕ is given by

$$\begin{aligned}
\frac{\partial \phi}{\partial t} &= \delta_\epsilon(\phi)\Big[\mu \nabla \cdot \left(\frac{\nabla \phi}{|\nabla \phi|}\right) - |z(x,y) - (a^+x + b^+y + c^+)|^2 \\
&+ |z(x,y) - (a^-x + b^-y + c^-)|^2 - \eta\Big((a^+)^2 + (b^+)^2\Big) + \eta\Big((a^-)^2 + (b^-)^2\Big)\Big].
\end{aligned}$$

Now we extend the idea of CV [35] to the two phase piecewise smooth approximation of the Mumford-Shah model [119]. We give some details of the piecewise smooth segmentation functional which is given below.

$$\begin{aligned}
\min_{\Gamma, s_1, s_2} F(\Gamma, s_1, s_2) &= \mu Length(\Gamma) + \int_{Inside(\Gamma)} ((z(x,y) - s_1(x,y))^2 + \eta|\nabla s_1(x,y)|^2)dxdy \\
&+ \int_{Outside(\Gamma)} ((z(x,y) - s_2(x,y))^2 + \eta|\nabla s_2(x,y)|^2)dxdy,
\end{aligned} \tag{3.37}$$

where $z(x,y)$ is a given image, s_1 and s_2 are two C^1 functions defined inside and outside the unknown contour Γ respectively, $\mu > 0$ controls the regularization of the length of the boundary of the smooth regions and $\eta > 0$ controls the regularization of the intensities of smooth regions. Now we consider the level set formulation of the functional as discussed in Section 3.5 and replacing the Heaviside function H by its regularized version H_ϵ, thus (3.37) becomes:

$$\begin{aligned}
\min_{\phi, s_1, s_2} F_\epsilon(\phi, s_1, s_2) &= \mu \int_\Omega |\nabla H_\epsilon(\phi)| + \int_\Omega ((z(x,y) - s_1(x,y))^2 + \eta|\nabla s_1(x,y)|^2 H_\epsilon(\phi)dxdy \\
&+ \int_\Omega ((z(x,y) - s_2(x,y))^2 + \eta|\nabla s_2(x,y)|^2(1 - H_\epsilon(\phi))dxdy.
\end{aligned} \tag{3.38}$$

Minimization leads to the following equations

$$\begin{aligned} s_1 &= z+\eta\Delta s_1 \text{ on } \{\phi>0\}, \frac{\partial s_1}{\partial \vec{n}} = 0 \text{ on } \{\phi=0\}\cup\partial\Omega, \\ s_2 &= z+\eta\Delta s_2 \text{ on } \{\phi<0\}, \frac{\partial s_2}{\partial \vec{n}} = 0 \text{ on } \{\phi=0\}\cup\partial\Omega, \\ \frac{\partial\phi}{\partial t} &= \delta_\epsilon(\phi)\left[\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right) - r_1(x,y)\right], \end{aligned} \tag{3.39}$$

where $r_1(x,y) = ((z(x,y)-s_1(x,y))^2 - (z(x,y)-s_2(x,y))^2 + \eta|\nabla s_1|^2 - \eta|\nabla s_2|^2)$.

3.7 Global Minimization of the Active Contour Model (CV2)

In this section we discuss the global minimization of the active contour without edges [35], proposed by T. F. Chan *et al.* [25]. Let us consider the 2-phase image segmentation model (CV2) (3.19)

$$\begin{aligned} &\min_{\phi,c_1,c_2} F(\phi,c_1,c_2) = \mu\int_\Omega |\nabla H(\phi)|dx + \nu\int_\Omega H(\phi)dx + \\ &\lambda_1\int_\Omega |z-c_1|^2 H(\phi)dx + \lambda_2\int_\Omega |z-c_2|^2(1-H(\phi))dx. \end{aligned} \tag{3.40}$$

This minimization problem is non-convex because of the length term $\int_\Omega |\nabla H(\phi)|dxdy$ [25]. Hence the optimization problem can have local minima, i.e the final segmented image might have wrong information. Despite of non-convex nature of (3.40), a natural way to a solution (ϕ, c_1, c_2) is a two step algorithm where c_1 and c_2 are firstly computed, and then look for best ϕ which minimize the energy $F(\phi, c_1, c_2)$. The gradient descent equation is

$$\begin{aligned} \frac{\partial\phi}{\partial t} &= \delta_\epsilon(\phi)\left[\mu\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right) - \underbrace{(\nu+\lambda_1(z-c_1)^2-\lambda_2(z-c_2)^2)}_{r(x,y)}\right] \\ \Rightarrow \frac{\partial\phi}{\partial t} &= \delta_\epsilon(\phi)\left[\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right) - \lambda r(x,y)\right]. \end{aligned} \tag{3.41}$$

In [35], the CV algorithm chooses a non-compactly supported, smooth approximation H_ϵ for H. Thus the above gradient descent equation and the one given below have the same stationary solutions:

$$\frac{\partial\phi}{\partial t} = \left[\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right) - \lambda r(x,y)\right], \tag{3.42}$$

where the $\delta_\epsilon(\phi)$ is dropped. The above equation (3.42) is the gradient descent equation of the following convex energy

$$\int_\Omega |\nabla\phi| + \lambda\int_\Omega r(x,y)\phi dxdy. \tag{3.43}$$

This energy is homogeneous of degree 1 in ϕ, so in general it does not have a minimizer, if we do not restrict the minimization to ϕ such as $0 \leqslant \phi \leqslant 1$ $\forall(x,y) \in \Omega$; see [25] for more details. Thus consider the following minimization problem

$$\min_{0\leqslant\phi(x,y)\leqslant1}\left(\int_\Omega|\nabla\phi|dxdy+\lambda\int_\Omega r(x,y)\phi dxdy\right),$$

and the claim is that this minimization leads to the global minimizer from the following theorem:

Theorem 3.7.1 *For any given fixed $c_1, c_2 \in \mathbb{R}$, a global minimizer for CV model can be found by carrying out the following convex minimization*

$$\min_{0\leqslant\phi(x,y)\leqslant1}\left(\int_\Omega|\nabla\phi|dxdy+\lambda\int_\Omega r(x,y)\phi dxdy\right). \tag{3.44}$$

The proof and further details can be found in [25]. To minimize (3.44), they have the following claim

Claim: Let $r(x,y) \in L^\infty(\Omega)$, then the convex, constrained minimization problem (3.44) has the same set of minimizers as the following convex, unconstrained minimization problem

$$\min_{\phi}\left(\int_\Omega|\nabla\phi|dxdy+\lambda\int_\Omega (r(x,y)\phi+\omega v(\phi))dxdy\right). \tag{3.45}$$

where $v(\zeta) := \max\{0, 2|\zeta-\frac{1}{2}|-1\}$, provided that $\omega > \frac{\lambda}{2}\|r(x,y)\|_{L^\infty(\Omega)}$. The proof can be found in [25]. The Euler Lagrange's equation is given by

$$\frac{\partial\phi}{\partial t}=\left[\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right)-\lambda r(x,y)\right]-\omega v'_{\epsilon_2}(\phi),$$

where v'_{ϵ_2} is the regularized version of v' with $v_{\epsilon_2}(\zeta)$ is given by [15]:

$$v_{\epsilon_2}(\zeta)=\begin{cases}-\zeta & \text{if } \zeta<-\epsilon_2/\sqrt{2},\\ (1+\sqrt{2})\zeta-\sqrt{tan^2(3\pi/8)\zeta^2-(\zeta-\epsilon_2)^2} & \text{if } -\epsilon_2/\sqrt{2}\leqslant\zeta<\epsilon_2,\\ 0 & \text{if } \epsilon_2\leqslant\zeta<1-\epsilon_2,\\ (1+\sqrt{2})\zeta-\sqrt{tan^2(3\pi/8)\zeta^2-(\zeta-1+\epsilon_2)^2} & \text{if } 1-\epsilon_2\leqslant\zeta<1+\epsilon_2/\sqrt{2},\\ \zeta-1 & \text{if } \zeta\geqslant1+\epsilon_2/\sqrt{2}.\end{cases} \tag{3.46}$$

An explicit scheme was used to solve this equation.

In [15], based on the energy (3.44), they proposed the following energy functional for minimization by introducing weighted TV

$$\min_{0\leqslant\phi(x,y)\leqslant1}\left(\int_\Omega g|\nabla\phi|dxdy+\lambda\int_\Omega r(x,y)\phi dxdy\right), \tag{3.47}$$

where g is an edge detector function and is defined as in (3.10) and $r(x,y) = (\nu+\lambda_1(z-c_1)^2-\lambda_2(z-c_2)^2)$. The minimization was carried out in similar way to [25]. Later on, they extended their discussion towards the global minimization of the active contour models based on the Mumford-Shah Model for the piecewise smooth case. To find the global minimizer of equation (3.39), same steps will be taken as discussed above, for more details see [15].

3.8 Geodesic Aided CV method

Geodesic active contours [21] are based on gradient to detect the boundary, which uses the local information. Firstly for images with fuzzy and discrete edges, it is difficult to get desirable results. Secondly, geodesic active contours are heavily affected by noisy inputs since the isotropic smoothing Gaussian must be strong, which will smooth the edges too. On the other hand, the CV method depends on the image information from homogenous regions which helps in detecting with fuzzy and discrete edges and also we do not need to de-noise the image, if the input image is noisy. The original CV method is unable to detect images with holes, i.e. the CV method has global minima deficiency. Also CV is based on the information of homogenous regions so the precise boundary may not be obtained. In [41], Li Chen et al introduced a new method to address the above mentioned problems by combining geodesic and CV methods. The CV PDE is

$$\frac{\partial \phi}{\partial t} = \delta_\epsilon(\phi)\left[\mu\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right) - \nu - \lambda_1(z(x,y)-c_1)^2 + \lambda_2(z(x,y)-c_2)^2\right],$$

where $z(x,y)$ is the given image, ϕ is the level set function and c_1 and c_2 are same as defined in Section 3.5. They made the following changes to the above partial differential equation. Firstly, they replace the delta term $\delta_\epsilon(\phi)$ by $|\nabla\phi|$, which then involves the whole image information. Secondly, the term $\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right)$ is replaced by $\nabla\cdot\left(g\frac{\nabla\phi}{|\nabla\phi|}\right)$, where g is the edge detector function defined in (3.10). Thus the new equation becomes

$$\frac{\partial \phi}{\partial t} = |\nabla\phi|\left[\mu\nabla\cdot\left(g\frac{\nabla\phi}{|\nabla\phi|}\right) - \nu - \lambda_1(z(x,y)-c_1)^2 + \lambda_2(z(x,y)-c_2)^2\right].$$

Since

$$\nabla\cdot\left(g\frac{\nabla\phi}{|\nabla\phi|}\right) = g\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right) + \nabla g\cdot\frac{\nabla\phi}{|\nabla\phi|},$$

after some manipulations and approximations they solve the following partial differential equation:

$$\frac{\partial \phi}{\partial t} = g|\nabla\phi|\left[\mu\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right) - \nu - \lambda_1(z(x,y)-c_1)^2 + \lambda_2(z(x,y)-c_2)^2\right] + \tau\nabla g\cdot\nabla z(x,y), \tag{3.48}$$

where they used $\tau = 1$, $\lambda_1 = \lambda_2 = 1$ and $\nu = 0$ [41]. Also they extended their method towards color images.

3.9 Conclusion

In this chapter we mainly discussed some variational models used for image segmentation also numerical methods used for solving the parabolic PDE arisen from the minimization of CV model. Geodesic active contours based on gradient and curvature to detect boundary, in which local information of boundary is used, cannot get ideal results when dealing with fuzzy edges and discrete edges. Furthermore because of the local attributes and the dependence on gradient, geodesic active contours are heavily affected by heavy noise. On the other hand, the CV method depends on the image information derived from homogenous regions, therefore it can obtain favorable results in fuzzy and discrete cases. De-noising process is also not necessary. The CV can detect objects in an image whose boundaries are not defined by gradient. Explicit scheme is popular in solving the parabolic PDEs, but due the Courant-Friedrichs-Lewy (CFL) condition which asserts that the numerical waves should propagate at least as fast as the physical waves [90], it requires very small time steps and therefore a large number of iterations required. Implicit scheme is stable for any time step, but for large time step the system matrix has a large condition number which causes slow convergence for iterative methods. The AOS scheme is also unconditionally stable like implicit scheme, for images of small sizes AOS is very efficient but very slow in convergence for images of large sizes. This motivates us to develop the multigrid method for solving the PDEs discussed in the next chapter.

Chapter 4

The Multigrid method for Active Contour without Edges

In this chapter we develop the multigrid method for the CV 2-phase image segmentation model, as introduced in chapter 3. We first work out multigrid method the piecewise constant model. Main references for this chapter are [8, 19, 22, 35, 50, 94, 95, 127, 130]

4.1 Introduction

Image segmentation is a central problem among image processing applications. The aim is to distinguish objects from background and to systematically select specific features out of an image that has many features [87, 6, 33]. For intensity-based images, the non-equation-based approaches are the most popular e.g threshold techniques, edge-based methods, region-based techniques, and connectivity-preserving relaxation method among others. One may also view the task of distinguishing objects of interest from "the rest" as one to identify the feature's boundaries. In recent years, a class of variational formulations offer us the ability to work out features with sharp boundaries – these are the new nonlinear approaches which require more sophisticated solution techniques [87, 33].

Let Ω be a bounded open subset of $\mathbb{R}^2$ with $\partial\Omega$ its boundary and let z be the given image, which may be a clean image or contains Gaussian noise. Our aim is to extract a desirable image u which represents features within z – more specifically u is piecewise constant inside each extracted feature.

In this chapter we present a working multigrid algorithm for implementing the Chan-Vese (CV) variational model [35] and to highlight the algorithm's practical advantages.

Section 4.2 first reviews related variational models and then describes the active contour without edges model by Chan and Vese [35], including a discussion of unilevel solution methods of semi-implicit and additive operator splitting. Section 4.3 first review the nonlinear multigrid framework and then describes our choice of smoothers as well as the multigrid algorithm for solving the underlying differential equation of [35]. Section 4.4 gives some local Fourier analysis of the smoothers used, which forms a basis for multigrid convergence. We end this chapter in Section 4.5 with some numerical results and conclusions.

4.2 The model of active contour without edges and solution methods

Variational segmentation methods aim to find edges of (piecewise smooth) features in an image by directly minimizing some objective functional. Several methods choose such functionals differently [33, 87]. In the sequel, we illustrate two early and related methods.

Firstly, the Mumford and Shah segmentation model [88]

$$\min_{u,K} F_1(u,K) = \int_{\Omega-K} (u-z)^2 dxdy + \alpha \int_{\Omega-K} |\nabla u|^2 dxdy + \beta \int_K d\sigma, \tag{4.1}$$

where α and β are nonnegative constants, z is the given image, u is the desired piecewise smooth image and $K \subset \Omega$ is the set of discontinuities, and $\int_K d\sigma$ is the length of K. This functional is clear but difficult to implement. Various attempts of approximating this formulation exist.

Secondly, the Ambrosio and Tortorelli model [4]

$$\min_{u,v} F_2(u,v) = \sigma \int_\Omega \left(\epsilon |\nabla v|^2 + \frac{(1-v)^2}{4\epsilon} \right) dxdy + \beta \int_\Omega v^2 |\nabla u|^2 dxdy + \lambda \int_\Omega (u-z)^2 dxdy \tag{4.2}$$

builds on the Modica and Mortola's Γ-convergence theory [85] of representing a two dimensional curve Γ by solving for the phase field function v $(0 \leqslant v \leqslant 1)$

$$\min_v \int_\Omega \left\{ \epsilon |\nabla v|^2 + \frac{(1-v)^2}{4\epsilon} \right\} dx.$$

Clearly model (4.2) appears more amenable to numerical implementation. It turns out that the above representation is a good approximation of $\int_K d\sigma$ shown in the Mumford and Shah model when ϵ is small [33, 109].

It is of interest to mention two other variational models. The snake model of [75] aims to find the segmentation curve C (a parameterized version of Γ with $C(s) : [0,1] \to \mathbb{R}^2$) by solving the problem

$$\min_C F_3(C) = \int_0^1 \left(\alpha |C'(s)|^2 + \beta |C''(s)| - \lambda |\nabla z(C(s))|^2 \right) ds, \tag{4.3}$$

where α, β and λ are positive parameters. The geodesic contour model of [21] proposes to find C by solving

$$\min_C F_4(C) = \int_0^1 |C'(s)|^2 g(|\nabla z(C(s))|) ds, \tag{4.4}$$

where g is an edge detection function e.g. for some $p \geqslant 1$ and a Gaussian $G_\sigma(x,y)$

$$g(|\nabla z(x,y)|) = \frac{1}{1 + |\nabla G_\sigma(x,y) * z(x,y)|^p}.$$

These models are discussed in chapter 3.

In this chapter our main focus is on developing multigrid method for solving PDE arisen from minimizing of CV model [35], which is already discussed in Section 3.5. Consider equation (3.23)

$$
\begin{aligned}
F_\epsilon(\phi, c_1, c_2) &= \mu \int_\Omega \delta_\epsilon(\phi(x,y))|\nabla\phi| dxdy + \lambda_1 \int_\Omega |z(x,y) - c_1|^2 H_\epsilon(\phi(x,y)) dxdy \\
&+ \lambda_2 \int_\Omega |z(x,y) - c_2|^2 (1 - H_\epsilon(\phi(x,y))) dxdy. \qquad (4.5)
\end{aligned}
$$

Minimization with respect to c_1, c_2 and ϕ we have the following equations

$$
\begin{cases}
c_1(\phi) = \dfrac{\int_\Omega z H_\epsilon(\phi) dxdy}{\int_\Omega H_\epsilon(\phi) dxdy}, \ c_2(\phi) = \dfrac{\int_\Omega z(1 - H_\epsilon(\phi)) dxdy}{\int_\Omega (1 - H_\epsilon(\phi)) dxdy}, \\
\delta_\epsilon(\phi)\Big[\mu\nabla.\Big(\frac{\nabla\phi}{|\nabla\phi|}\Big) - \lambda_1(z(x,y) - c_1)^2 + \lambda_2(z(x,y) - c_2)^2\Big] = 0, \qquad \text{in } \Omega, \\
\phi(0,x,y) = \phi_0(x,y) \qquad (\text{ will be given}), \qquad \text{in } \Omega, \\
\dfrac{\delta_\epsilon(\phi)}{|\nabla\phi|}\dfrac{\partial\phi}{\partial n} = 0, \qquad \text{on } \partial\Omega.
\end{cases} \qquad (4.6)
$$

For given ϕ we compute c_1 and c_2 and keep them fixed and solve the non linear PDE in equation (4.6) for ϕ using multigrid method, Once ϕ is updated we use new ϕ to compute c_1 and c_2 and so on. Details are given in next section.

4.3 A nonlinear multi-grid method

Our main focus in this chapter is to present a multigrid algorithm for solving the nonlinear PDE in equation (4.6) directly without involving artificial time steps (as done when using time marching methods). We are not aware of similar work done for segmentation models in the level set formulation. For image restoration models, there exist multigrid algorithms [24, 105].

Although without using the artificial variable t, we also denote the approximation at (i,j) by $\phi_{i,j} = \phi(x_i, y_j)$. Let us consider the elliptic PDE:

$$
\begin{aligned}
\delta_\epsilon(\phi)\Big[\mu \mathrm{div}\Big(\frac{\nabla\phi}{|\nabla\phi|}\Big) - \lambda_1(z(x,y) - c_1)^2 + \lambda_2(z(x,y) - c_2)^2\Big] &= 0, \\
\mu \mathrm{div}\Big(\frac{\nabla\phi}{|\nabla\phi|}\Big) - \lambda_1(z(x,y) - c_1)^2 + \lambda_2(z(x,y) - c_2)^2 &= 0. \qquad (4.7)
\end{aligned}
$$

Here equation (4.7), is Euler-Lagrange's equation of the functional

$$
\mu \int_\Omega |\nabla\phi| dxdy + \int_\Omega (\lambda_1(z(x,y) - c_1)^2 - \lambda_2(z(x,y) - c_2)^2)\phi(x,y) dxdy. \qquad (4.8)
$$

Equation (4.8) and (4.5) have the same stationary points [25, 15]. But equation (4.8) is homogenous in ϕ of degree 1, [25]. This means that this evolution equation does not have stationary solution, so need to restrict the minimization to ϕ such that $0 \leqslant |\phi| \leqslant 1$.

Using finite difference scheme to discretize the equation (4.7) for ϕ. The equation at a grid point (i,j) is given by

$$
\begin{aligned}
&\Big[\mu\Big\{\frac{\Delta_-^x}{h_1}\Big(\frac{\Delta_+^x \phi_{i,j}/h_1}{\sqrt{(\Delta_+^x \phi_{i,j}/h_1)^2 + (\Delta_+^y \phi_{i,j}/h_2)^2 + \beta}}\Big) \\
&+ \frac{\Delta_-^y}{h_2}\Big(\frac{\Delta_+^y \phi_{i,j}/h_2}{\sqrt{(\Delta_+^x \phi_{i,j}/h_1)^2 + (\Delta_+^y \phi_{i,j}/h_2)^2 + \beta}}\Big)\Big\} - \lambda_1(z_{i,j} - c_1)^2 + \lambda_2(z_{i,j} - c_2)^2\Big] = 0,
\end{aligned}
$$

where $\beta > 0$ is a small parameter to avoid zero denominator.

Remark 4.3.1 *We have used different values of $\beta \in (0, 1]$, it has no effect on the final results.*

The above equation (4.9) implies that

$$
\begin{aligned}
&\Big[\underline{\mu}\Big\{\Delta^x_-\Big(\frac{\Delta^x_+\phi_{i,j}}{\sqrt{(\Delta^x_+\phi_{i,j})^2+(\lambda\Delta^y_+\phi_{i,j})^2+\bar{\beta}}}\Big)+\lambda^2\Delta^y_-\Big(\frac{\Delta^y_+\phi_{i,j}}{\sqrt{(\Delta^x_+\phi_{i,j})^2+(\lambda\Delta^y_+\phi_{i,j})^2+\bar{\beta}}}\Big)\Big\}\\
&- \ \lambda_1(z_{i,j}-c_1)^2+\lambda_2(z_{i,j}-c_2)^2=0,
\end{aligned}
$$

$$
\begin{aligned}
\Longrightarrow \quad &\underline{\mu}\Big\{\Delta^x_-\Big(\frac{\Delta^x_+\phi_{i,j}}{\sqrt{(\Delta^x_+\phi_{i,j})^2+(\lambda\Delta^y_+\phi_{i,j})^2+\bar{\beta}}}\Big)+\lambda^2\Delta^y_-\Big(\frac{\Delta^y_+\phi_{i,j}}{\sqrt{(\Delta^x_+\phi_{i,j})^2+(\lambda\Delta^y_+\phi_{i,j})^2+\bar{\beta}}}\Big)\Big\}\\
&= \lambda_1(z_{i,j}-c_1)^2-\lambda_2(z_{i,j}-c_2)^2, \qquad (4.9)
\end{aligned}
$$

where $\underline{\mu} = \mu/h_1$, $\bar{\beta} = h_1^2\beta$ and $\lambda = h_1/h_2$, with Neumann's boundary conditions

$$
\begin{gathered}
\phi_{i,0}=\phi_{i,1}, \quad \phi_{i,m_2+1}=\phi_{i,m_2}, \quad \phi_{0,j}=\phi_{1,j}, \quad \phi_{m_1+1,j}=\phi_{m_1,j}, \qquad (4.10)\\
\text{for } i=1,\dots,m_1, \quad j=1,\dots,m_2 \text{ and } 0\leqslant|\phi_{i,j}|\leqslant 1.
\end{gathered}
$$

Where in Eq. (4.9) left hand side resembles the denoising model by [99] using the total variation (TV) regularization. The parameter β should be a small quantity to avoid the gradient becoming 0 as in [99, 122, 123]

4.3.1 The full approximation scheme

We first give a brief discussion of the 3 main ingredients of a nonlinear multigrid (MG) called the full approximation scheme (FAS) [68, 39, 24, 118], due to A. Brandt [14], and then concentrate on our choice of smoothers. Denote the system of non linear equations described by equation (4.9) and (4.10) by

$$N^h(\phi^h) = f^h \qquad (4.11)$$

where $h_1 = h_2 = h$, ϕ^h and f^h are grid functions on a $m_1 \times m_2$ cell centered rectangular grid Ω^h with spacing (h_1, h_2). Let Ω^{2h} denote the $m_1/2 \times m_2/2$ cell centered grid which results from standard coarsening of Ω^h. Let $e^h = \phi^h - \Phi^h$ be the solution error, where Φ^h is a good approximation to solution of (4.11) in the sense that e^h is smooth. Such smoothness can only be achieved by a careful choice of suitable smoothers – a major task in developing a working multigrid method.

Let $r^h = f^h - N^h(\Phi^h)$ be the residual. Then the non-linear residual equation will be:

$$N^h(\Phi^h + e^h) - N^h(\Phi^h) = r^h. \qquad (4.12)$$

If e^h is smooth, it can be well approximated on Ω^{2h}. Therefore any iterative method which smooths the error on the fine grid can be improved by the use of the coarse grid correction, in which a coarse grid analogue of the residual equation is solved (the solution on a coarse grid is less expensive than on a fine grid) to obtain a coarse grid approximation of the error, which is then transferred back to the fine grid to correct the approximation Φ^h. This is known as a two-grid cycle, and with recursive application can be extended to a multigrid method. Let us define the restriction and interpolation operators for transferring grid functions between Ω^h and Ω^{2h} for cell-centered discretization:

Restriction

$$I_h^{2h}\Phi^h = \Phi^{2h}$$

where

$$\Phi_{i,j}^{2h} = \frac{1}{4}(\Phi_{2i-1,2j-1}^{h} + \Phi_{2i-1,2j}^{h} + \Phi_{2i,2j-1}^{h} + \Phi_{2i,2j}^{h}), \qquad 1 \leqslant i \leqslant m_1/2, \quad 1 \leqslant j \leqslant m_2/2.$$

is a full weighting operator [39, 118].

Interpolation

$$I_{2h}^{h}\Phi^{2h} = \Phi^{h}$$

where

$$\begin{aligned}
\Phi_{2i,2j}^{h} &= \frac{1}{16}(9\Phi_{i,j}^{2h} + 3\Phi_{i+1,j}^{2h} + 3\Phi_{i,j+1}^{2h} + \Phi_{i+1,j+1}^{2h}),\\
\Phi_{2i-1,2j}^{h} &= \frac{1}{16}(9\Phi_{i,j}^{2h} + 3\Phi_{i-1,j}^{2h} + 3\Phi_{i,j+1}^{2h} + \Phi_{i-1,j+1}^{2h}),\\
\Phi_{2i,2j-1}^{h} &= \frac{1}{16}(9\Phi_{i,j}^{2h} + 3\Phi_{i+1,j}^{2h} + 3\Phi_{i,j-1}^{2h} + \Phi_{i+1,j-1}^{2h}),\\
\Phi_{2i-1,2j-1}^{h} &= \frac{1}{16}(9\Phi_{i,j}^{2h} + 3\Phi_{i-1,j}^{2h} + 3\Phi_{i,j-1}^{2h} + \Phi_{i-1,j-1}^{2h}),\\
&\text{for} \qquad 1 \leqslant i \leqslant m_1/2, \qquad 1 \leqslant j \leqslant m_2/2.
\end{aligned}$$

is known as a bilinear interpolation operator [39, 118].

It remains to discuss the most important ingredient of a MG: smoothing. We will first discuss a local nonlinear smoother and then review the smoother introduced by Savage and Chen [105].

4.3.2 Smoother I: Local Smoother

In this method the system of nonlinear equations is linearized locally, by computing $D(\phi)$ on each grid (i,j) locally. Then we get a system of linear equations. As the Gauss Seidel has a best smoothing property, we apply the Gauss-Seidel method to this system of linear equations to smooth the error. We are using a few steps of this smoother to smooth the error which ensure a convergent nonlinear multigrid. Equation (4.9) can be written as

$$\begin{aligned}
&\underline{\mu}\Big\{\Big[\frac{\Delta_+^x\phi_{i,j}}{\sqrt{(\Delta_+^x\phi_{i,j})^2 + (\lambda\Delta_+^y\phi_{i,j})^2 + \bar{\beta}}} - \frac{\Delta_+^x\phi_{i-1,j}}{\sqrt{(\Delta_+^x\phi_{i-1,j})^2 + (\lambda\Delta_+^y\phi_{i-1,j})^2 + \bar{\beta}}}\Big]\\
&+\lambda^2\Big[\frac{\Delta_+^y\phi_{i,j}}{\sqrt{(\Delta_+^x\phi_{i,j})^2 + (\lambda\Delta_+^y\phi_{i,j})^2 + \bar{\beta}}} - \frac{\Delta_+^y\phi_{i,j-1}}{\sqrt{(\Delta_+^x\phi_{i,j-1})^2 + (\lambda\Delta_+^y\phi_{i,j-1})^2 + \bar{\beta}}}\Big]\Big\}\\
&\qquad = \lambda_1(z_{i,j} - c_1)^2 - \lambda_2(z_{i,j} - c_2)^2.
\end{aligned}$$

Let the coefficients (intended below to be frozen in local linearization) be denoted by

$$\begin{aligned}
D(\phi)_{i,j} &= \frac{1}{\sqrt{(\Delta_+^x\phi_{i,j})^2 + (\lambda\Delta_+^y\phi_{i,j})^2 + \bar{\beta}}},\\
D(\phi)_{i-1,j} &= \frac{1}{\sqrt{(\Delta_+^x\phi_{i-1,j})^2 + (\lambda\Delta_+^y\phi_{i-1,j})^2 + \bar{\beta}}},\\
D(\phi)_{i,j-1} &= \frac{1}{\sqrt{(\Delta_+^x\phi_{i,j-1})^2 + (\lambda\Delta_+^y\phi_{i,j-1})^2 + \bar{\beta}}},
\end{aligned}$$

so we have

$$\begin{aligned}
&\underline{\mu}\{(D(\phi)_{i,j}\Delta_+^x\phi_{i,j} - D(\phi)_{i-1,j}\Delta_+^x\phi_{i-1,j}) + \lambda^2(D(\phi)_{i,j}\Delta_+^y\phi_{i,j} - D(\phi)_{i,j-1}\Delta_+^x\phi_{i,j-1})\}\\
&\qquad = \lambda_1(z_{i,j} - c_1)^2 - \lambda_2(z_{i,j} - c_2)^2,
\end{aligned}$$

which implies

$$\begin{aligned}\underline{\mu}\Big\{&\Big[D(\phi)_{i,j}(\phi_{i+1,j}-\phi_{i,j})-D(\phi)_{i-1,j}(\phi_{i,j}-\phi_{i-1,j})\Big]+\\ &\lambda^2\Big[D(\phi)_{i,j}(\phi_{i,j+1}-\phi_{i,j})-D(\phi)_{i,j-1}(\phi_{i,j}-\phi_{i,j-1})\Big]\Big\}\\ &\qquad=\lambda_1(z_{i,j}-c_1)^2-\lambda_2(z_{i,j}-c_2)^2=f_{i,j}.\end{aligned}\tag{4.13}$$

Clearly $D(\phi)_{i,j}$, $D(\phi)_{i-1,j}$ and $D(\phi)_{i,j-1}$ all contains $\phi_{i,j}$ terms, which will be evaluated at past iterations in a coefficient-freezing process. Let $\widetilde{\phi}$ be approximation to ϕ. By putting the values of $\widetilde{\phi}$ at each grid point in equation (4.13) other than the grid point (i,j) and also finding the value of D at each grid point (i,j), we get a linear equation in one variable $\phi_{i,j}$,

$$\begin{aligned}\underline{\mu}\Big\{&\Big[D(\widetilde{\phi})_{i,j}(\widetilde{\phi}_{i+1,j}-\phi_{i,j})-D(\widetilde{\phi})_{i-1,j}(\phi_{i,j}-\widetilde{\phi}_{i-1,j})\Big]+\\ &\lambda^2\Big[D(\widetilde{\phi})_{i,j}(\widetilde{\phi}_{i,j+1}-\phi_{i,j})-D(\widetilde{\phi})_{i,j-1}(\phi_{i,j}-\widetilde{\phi}_{i,j-1})\Big]\Big\}\equiv f_{i,j}\end{aligned}$$

implies

$$\begin{aligned}\Big\{&\Big[D(\widetilde{\phi})_{i,j}(\widetilde{\phi}_{i+1,j}-\phi_{i,j})-D(\widetilde{\phi})_{i-1,j}(\phi_{i,j}-\widetilde{\phi}_{i-1,j})\Big]+\\ &\lambda^2\Big[D(\widetilde{\phi})_{i,j}(\widetilde{\phi}_{i,j+1}-\phi_{i,j})-D(\widetilde{\phi})_{i,j-1}(\phi_{i,j}-\widetilde{\phi}_{i,j-1})\Big]\Big\}=f_{i,j}/\underline{\mu}\equiv\bar{f}_{i,j}.\end{aligned}$$

Our proposed algorithm solves this equation for $\phi_{i,j}$ to update the approximation at each pixel (i,j):

Algorithm 6 (Algorithm for first smoother)

$$\Phi^h \longleftarrow Smoother1(\Phi^h, \bar{f}^h, maxit, tol)$$

```
where maxit is the maximum no of inner iterations.
for $i = 1 : m_1$
    for $j = 1 : m_2$
        for iter=1:maxit
        $\widetilde{\Phi}^h \leftarrow \Phi^h$
$\Phi_{i,j} = \dfrac{\Big[\Big\{D(\widetilde{\Phi}^h)_{i,j}\widetilde{\Phi}^h_{i+1,j}+D(\widetilde{\Phi}^h)_{i-1,j}\widetilde{\Phi}^h_{i-1,j}+\lambda^2 D(\widetilde{\Phi}^h)_{i,j}\widetilde{\Phi}^h_{i,j+1}+\lambda^2 D(\widetilde{\Phi}^h)_{i,j-1}\widetilde{\Phi}^h_{i,j-1}\Big\}-\bar{f}_{i,j}\Big]}{D(\widetilde{\Phi}^h)_{i,j}+D(\widetilde{\Phi}^h)_{i-1,j}+\lambda^2(D(\widetilde{\Phi}^h)_{i,j}+D(\widetilde{\Phi}^h)_{i,j-1})}$
        if $|\Phi_{i,j} - \widetilde{\Phi}^h_{i,j}| < tol$ then stop
        end
    end
end
```

4.3.3 Smoother II: Global Smoother

We now review the smoother proposed in [105] for a different image model. In this method the system of non linear equations is linearized globally at each step by computing $D(\phi)$ on each grid point (i,j). To the resulting system of linear equations Gauss-Seidel relaxation is applied. This global smoother is different from the local smoother defined above. The algorithm proceeds as follows:

Algorithm 7 (Algorithm for second smoother)

$$\Phi^h \longleftarrow Smoother2(\Phi^h, \bar{f}^h, maxit, tol)$$

```
for $i = 1 : m_1$
    for $j = 1 : m_2$
        $D(\Phi^h)_{i,j} = [(\Delta_+^x \Phi_{i,j})^2 + (\lambda \Delta_+^y \Phi_{i,j})^2 + \bar{\beta}]^{-\frac{1}{2}}$
    end
end
$\varphi^h = \Phi^h$
for $iter = 1 : maxit$
    for $i = 1 : n$
        for $j = 1 : m$
        $\widetilde{\varphi}^h \leftarrow \varphi^h$
$\varphi_{i,j} = \frac{\left[\left\{D(\Phi^h)_{i,j}\widetilde{\varphi}^h_{i+1,j} + D(\Phi^h)_{i-1,j}\widetilde{\varphi}^h_{i-1,j} + \lambda^2 D(\Phi^h)_{i,j}\widetilde{\varphi}^h_{i,j+1} + \lambda^2 D(\Phi^h)_{i,j-1}\widetilde{\varphi}^h_{i,j-1}\right\} - \bar{f}_{i,j}\right]}{D(\Phi^h)_{i,j} + D(\Phi^h)_{i-1,j} + \lambda^2 (D(\Phi^h)_{i,j} + D(\Phi^h)_{i,j-1})}$
        end
    end
end
$\Phi^h \leftarrow \varphi$
```

Here updating to the coefficients needs to be done at the start of each smoothing step globally and to be stored for relaxation use. This was found to be necessary for the total-variation denoising model of [99].

Remark 4.3.2 *The above smoothers are both fixed point based. Then one may consider two related ideas: (i) Newton methods – which are found not to perform satisfactorily for this problem and also previously for the TV denoising problem [105]. (ii) Line relaxation method – which are found to work well but the improvements over smoothers I and II are marginal (of course line relaxations are slightly more expensive to implement).*

Yet there exist other smoothing ideas in the literature, e.g., the energy minimizing smoother of [124], the primal-dual smoother [31] and algebraic multigrid ideas [37, 38, 101] which remain to be tested for segmentation problems.

4.3.4 The multi-grid algorithm

To solve equation (4.11) by the multigrid method, the algorithm is as follows, see [39, 118] and references therein :

Algorithm 8 (Multigrid Algorithm) *Assume we have set up these multigrid parameters:*

ν_1 *pre-smoothing steps on each level*
ν_2 *post-smoothing steps on each level*
γ *the number of multigrid cycles on each level ($\gamma = 1$ for V-cycling and $\gamma = 2$ for W-cycling).*
rr: *Relative residual.*
For given Φ^h compute $\bar{f}^h$ and keep it fixed. Here we present one step V-cycle of nonlinear

multigrid method for CV model.
FAS Multigrid Cycle
Start

$$\Phi^h \longleftarrow FASCYC(\Phi^h, \bar{f}^h, iter, \nu_1, \nu_2, \gamma, tol)$$

$\Phi_0 = \Phi^h$

1. *If we are on the coarsest grid i.e Ω^h is coarsest grid, then solve equation (4.11) using time marching technique of [127] and then stop.*
   Else use the smoother i.e

   $$\Phi^h \longleftarrow Smoother^{\nu_1}(\Phi^h, \bar{f}^h, iter, \nu_1, \nu_2, \gamma). \qquad \textit{(Pre-Smoothing)}$$

2. **Restriction:**

   $$\Phi^{2h} = I_h^{2h}\Phi^h, \quad \bar{\Phi}^{2h} = \Phi^{2h}.$$
   $$\bar{f}^{2h} = I_h^{2h}(\bar{f}^h - H^h\Phi^h) + N^{2h}(\Phi^{2h})$$
   $$\Phi^{2h} \longleftarrow FASCYC_\gamma^{2h}(\Phi^h, \bar{f}^h, iter, \nu_1, \nu_2, \gamma)$$

3. **Interpolation**

   $$\Phi^h \longleftarrow \Phi^h + I_{2h}^h(\Phi^{2h} - \bar{\Phi}^{2h})$$

4. $$\Phi^h \longleftarrow Smoother^{\nu_2}(\Phi^h, \bar{f}^h, iter, \nu_1, \nu_2, \gamma). \qquad \textit{(Post-Smoothing)}$$

   Update $\bar{f}^h$

   If $\mathbf{rr} = \dfrac{\|\Phi^h - \Phi_0\|_2}{\|\Phi_0\|_2} < tol$ *Stop.*

   Else go to **Start**.

4.4 Local Fourier analysis of smoothers

The standard FAS multilevel algorithm (such as Algorithm 8) does not automatically converge for many problems, if simple smoothers are used (i.e. Gauss-Seidel for linear problems and Gauss-Seidel-Newton for nonlinear problems). The key for convergence lies in effective smoothers or reduction of residuals to a smoothed form (where high frequency components are small regardless of the overall error [39, 118]). Here we show some local Fourier analysis (LFA) results to suggest that our smoothers are effective.

It should be remarked that LFA is in general not applicable to nonlinear smoothers. Here for our linearized smoothers, the analysis can only be done for each individual smoothing iteration and the obtained smoothing rates change from iteration to iteration. However, we look for general trends e.g. if the three consecutive smoothing rates are 0.59,0.61,0.44 (instead of a constant rate say 0.5), we say the underlying smoother is effective. Likewise, consecutive rates such as 1.2,0.89,0.99 may indicate a poor smoother.

For simplicity, we consider the case of a square image $m = m_1 = m_2$. Denote $h = h_1 = h_2$. The typical grid equation on Ω^h is

$$D(\phi_{i,j})(\phi_{i+1,j} - \phi_{i,j}) - D(\phi_{i-1,j})(\phi_{i,j} - \phi_{i-1,j}) +$$
$$\lambda^2[D(\phi_{i,j})(\phi_{i,j+1} - \phi_{i,j}) - D(\phi_{i,j-1})(\phi_{i,j} - \phi_{i,j-1})] = \bar{f}_{i,j}.$$

For the local smoother let $g_1 = D(\widetilde{\phi})_{i-1,j} = D(\phi^{(k)})_{i-1,j}$, $g_2 = D(\widetilde{\phi})_{i,j} = D(\phi^{(k)})_{i,j}$ and $g_3 = D(\widetilde{\phi})_{i,j-1} = D(\phi^{(k)})_{i,j-1}$, and for the global smoother $g_1 = D(\widetilde{\Phi})_{i-1,j}$, $g_2 = D(\widetilde{\Phi})_{i,j}$ and $g_3 = D(\widetilde{\Phi})_{i,j-1}$ where $\widetilde{\Phi}$ is the iterate at the previous sweep (global fixed-point). Also as $h_1 = h_2$, we have $\lambda^2 = 1$. Thus we obtain

$$-(g_1 + 2g_2 + g_3)\phi_{i,j} + g_1\phi_{i-1,j} + g_3\phi_{i,j-1} + g_2(\phi_{i,j+1} + \phi_{i,j+1}) = \bar{f}_{i,j},$$

or

$$-(g_1 + 2g_2 + g_3)\phi_{i,j}^{k+1} + g_1\phi_{i-1,j}^{k+1} + g_3\phi_{i,j-1}^{k+1} + g_2(\phi_{i,j+1}^{k} + \phi_{i+1,j}^{k}) = \bar{f}_{i,j}.$$

Now the exact solution $\phi_{i,j}$ will satisfy the discretized equation

$$-(g_1 + 2g_2 + g_3)\phi_{i,j} + g_1\phi_{i-1,j} + g_3\phi_{i,j-1} + g_2(\phi_{i,j+1} + \phi_{i+1,j}) = \bar{f}_{i,j},$$

as $\bar{f}_{i,j}$ is fixed thus we have

$$-(g_1 + 2g_2 + g_3)e_{i,j}^{k+1} + g_1 e_{i-1,j}^{k+1} + g_3 e_{i,j-1}^{k+1} + g_2(e_{i,j+1}^{k} + e_{i+1,j}^{k}) = 0, \qquad (4.14)$$

where $e_{i,j}^{k+1} = \phi_{i,j} - \phi_{i,j}^{k+1}$ and $e_{i,j}^{k} = \phi_{i,j} - \phi_{i,j}^{k}$ are the local error functions after and before the pre(post) smoothing step respectively.

Recall that the local Fourier analysis (LFA) measures the largest amplification factor in a relaxation scheme [14, 39, 118]. Let the general Fourier component be

$$B_{\theta_1,\theta_2}(x_i, y_j) = \exp\left(\mathrm{i}\alpha_1\frac{x_i}{h} + \mathrm{i}\alpha_2\frac{y_j}{h}\right) = \exp\left(\frac{2\mathrm{i}\theta_1 i\pi}{m} + \frac{2\mathrm{i}\theta_2 j\pi}{m}\right), \quad \mathrm{i} = \sqrt{-1}.$$

Here $\alpha_1 = \dfrac{2\theta_1\pi}{m}, \alpha_2 = \dfrac{2\theta_2\pi}{m} \in [-\pi, \pi]$. The LFA involves expanding

$$e^{k+1} = \sum_{\theta_1,\theta_2=-m/2}^{m/2} \psi_{\theta_1,\theta_2}^{k+1} B_{\theta_1,\theta_2}(x_i, y_j), \quad e^{k} = \sum_{\theta_1,\theta_2=-m/2}^{m/2} \psi_{\theta_1,\theta_2}^{k} B_{\theta_1,\theta_2}(x_i, y_j)$$

in Fourier components. We shall estimate the maximum ratio

$$\bar{\mu} = \max_{\theta_1,\theta_2} \mu(\theta_1, \theta_2) = |\psi_{\theta_1,\theta_2}^{k+1} / \psi_{\theta_1,\theta_2}^{k}|$$

in the high frequency range $(\alpha_1, \alpha_2) \in [-\pi, \pi] \setminus [\frac{-\pi}{2}, \frac{\pi}{2}]$ which defines the smoothing rate [118]. Now we replace all grid functions by their Fourier series and essentially consider the so called amplification factor i.e. the ratio between ψ_θ^{k+1} and ψ_θ^{k} for each θ where $\theta = (\theta_1, \theta_2)$. Then for the Fourier component of the error functions $e_{i,j}^{k}$ and $e_{i,j}^{k+1}$ before and after relaxation sweep, let us consider

$$e_{i,j}^{k} = \psi_\theta^{k} e^{\mathrm{i}(2\pi\theta_1 i + 2\pi\theta_2 j)/m} \quad \mathit{and} \quad e_{i,j}^{k+1} = \psi_\theta^{k+1} e^{\mathrm{i}(2\pi\theta_1 i + 2\pi\theta_2 j)/m}, \qquad (4.15)$$

so from equation (4.14) we have

$$\begin{aligned} & -(g_1 + 2g_2 + g_3)\psi_\theta^{k+1} e^{\mathrm{i}(2i\pi\theta_1 + 2j\pi\theta_2)/m} + g_1\psi_\theta^{k+1} e^{\mathrm{i}(2\pi(i-1)\theta_1 + 2\pi j\theta_2)/m} \\ + \quad & g_3\psi_\theta^{k+1} e^{\mathrm{i}(2\pi i\theta_1 + 2\pi(j-1)\theta_2)/m} \\ + \quad & g_2\Big[\psi_\theta^{k} e^{\mathrm{i}(2\pi i\theta_1 + 2\pi(j+1)\theta_2)/m} + \psi_\theta^{k} e^{\mathrm{i}(2\pi(l+1)\theta_1 + 2\pi j\theta_2)/m}\Big] = 0. \end{aligned} \qquad (4.16)$$

implies that

$$\Big[g_1 + 2g_2 + g_3 - g_1 e^{-\mathrm{i}\theta_1} - g_3 e^{-\mathrm{i}\theta_2}\Big]\psi_\theta^{k+1} = g_2 \psi_\theta^k \Big[e^{\mathrm{i}\theta_1} + e^{\mathrm{i}\theta_2}\Big].$$

$$\begin{aligned}
\Longrightarrow \quad \mu(\theta) = \Big|\frac{\psi_\theta^{k+1}}{\psi_\theta^k}\Big| &= \Big|\frac{g_2(e^{\mathrm{i}\theta_1} + e^{\mathrm{i}\theta_2})}{(g_1 + 2g_2 + g_3 - g_1 e^{-\mathrm{i}\theta_1} - g_3 e^{-\mathrm{i}\theta_2})}\Big| \\
= \quad & \frac{|g_2(e^{\mathrm{i}\theta_1} + e^{\mathrm{i}\theta_2})|}{|g_1 + 2g_2 + g_3 - g_1 e^{-\mathrm{i}\theta_1} - g_3 e^{-\mathrm{i}\theta_2}|} = \frac{A}{B},
\end{aligned} \tag{4.17}$$

where

$$\begin{aligned}
A &= |g_2(e^{\mathrm{i}\theta} + e^{\mathrm{i}\theta})| = |g_2||(\cos\theta_1 + \cos\theta_2) + \mathrm{i}(\sin\theta_1 + \sin\theta_2)| \\
&= 2\sqrt{|g_2|}\cos(\frac{\theta_1 - \theta_2}{2}) \quad \textit{and} \\
B &= |g_1 + 2g_{+}g_3 - g_1 e^{-\mathrm{i}\theta_1} - e^{-\mathrm{i}\theta_2}| \\
&= |[(g_1 + 2g_2 + g_3) - (g_1 \cos\theta_1 + \cos\theta_2)] + \mathrm{i}[g_1 \sin\theta_1 + g_3 \sin\theta_2]| \\
&= 2\sqrt{g_1(g_1 + 2g_2)\sin^2\frac{\theta_1}{2} + g_3(g_3 + 2g_2)\sin^2\frac{\theta_2}{2} + g_2^2 + 2g_1 g_3 \cos(\frac{\theta_1 - \theta_2}{2})\sin\frac{\theta_1}{2}\sin\frac{\theta_2}{2}}.
\end{aligned}$$

Define $|\theta| = max(|\theta_1|, |\theta_2|)$. Here we are looking for smoothing factor $\bar{\mu}$, which is given by

$$\bar{\mu} = \max_{\hat{\rho}\pi \leqslant |\theta| \leqslant \pi} \mu(\theta),$$

where $\hat{\rho}$ is the mesh size ratio and the range $\hat{\rho}\pi \leqslant |\theta| \leqslant \pi$ is the suitable range of high frequency components, i.e. the range of components that cannot be approximated on the coarser grid. We are using standard coarsening so $\hat{\rho} = \frac{1}{2}$, [14]. We will present smoothing factor $\bar{\mu}$ for both smoothers. To proceed with an analysis, we have to compute g_1, g_2 and g_3 or the function

$$D(\phi) = \sqrt{(\Delta_+^x \phi)^2 + (\Delta_+^y \phi)^2 + \bar{\beta}},$$

numerically and work out the smoothing factor $\bar{\mu}$ for each set of coefficients g_1, g_2 and g_3 within a smoother. In our earlier work, we select a special set of such coefficients. We use the complete set of coefficients g_1, g_2 and g_3 to compute the smoothing factors $\bar{\mu}$. We display the maximum of such factors

$$\hat{\mu} = \max_{g_1, g_2, g_3} \bar{\mu} = \max_{g_1, g_2, g_3} \max_{\theta} \mu(\theta).$$

As such a linear analysis is based on freezing the nonlinear coefficients, our results should be viewed only as a guide to smoother's effectiveness and a way to distinguish smoothers.

Taking the test example of first image of third row from Fig. 3.1 page 60 with $m = 32$, i.e. reducing the size to 32×32, we can display $\hat{\mu}$ in the first 4 cycles of our MG algorithm as in Table 4.1 where Pre-1 refers to the case of "pre-smoothing" and Post-1 to "post-smoothing" etc. If we instead consider the average rate from all pixels, the averages are respectively 0.49 and 0.71 for Smoothers I and II. Clearly in this example Smoother I appears to be more effective than Smoother II in terms of rates. Such a claim will be tested in the numerical results section.

Table 4.1: $\hat{\mu}$ in the first 4 cycles of out MG algorithm.

MG cycle	Smoothing Steps	Rate I:$\hat{\mu}^I$	Rate II:$\hat{\mu}^{II}$
1	Pre-1	0.4942	0.6776
	Pre-2	0.4941	0.9317
	Post-1	0.4942	0.9135
	Post-2	0.4942	0.9427
2	Pre-1	0.6003	0.9561
	Pre-2	0.6003	0.9174
	Post-1	0.6003	0.9581
	Post-2	0.6003	0.9577
3	Pre-1	0.7760	0.9533
	Pre-2	0.7760	0.9193
	Post-1	0.7757	0.9092
	Post-2	0.7749	0.9040
4	Pre-1	0.6025	0.9594
	Pre-2	0.6026	0.9456
	Post-1	0.6026	0.9286
	Post-2	0.6026	0.9678

4.5 Numerical results

In this section, we shall discuss several aspects of multigrid method. First we demonstrate that using multigrid method we can get the improved solution for the global minimizer. Secondly, we illustrate the fast convergence of the method by presenting the relative residuals and CPU time with both smoothers. Lastly, we give a comparison of multigrid method with Additive Operator Splitting (AOS) and Semi Implicit (SI) methods by giving there CPU times.

4.5.1 Results using smoother I

This section gives some experimental results of our MG algorithm 8 with smoother I (local smoother). In Fig. 4.1 our MG algorithm is tested on an synthetic noisy image having different objects. The parameters used in this test are $\mu = 0.1 \times (256)^2$ and $\lambda_1 = \lambda_2 = 1$ are same for all images. Also $\beta = 10^{-6}, \epsilon = 1 \times 10^{-20}, \phi_0 = -\sqrt{(x-150)^2 + (y-140)^2} + 100$. The top left figure is the original image with initial contour and top right is the initial segmented image. The 2nd row gives the results after one cycle [1] and the 3rd row presents the final results with 3 cycles with relative residual[2] rr $= 1.246 \times 10^{-15}$. For these experiments we used Matlab programming. The parameter ϵ can be set zero on fine levels as on fine levels H_ϵ is used to compute c_1 and c_2. But on coarsest level ϵ can not be zero. The chosen value of ϵ was found experimentally to be optimal for our images. In Fig. 4.2, our MG algorithm is tested on an artificial noisy image to see if the inner boundary of object could be detected. Clearly, the inner boundary of the object is detected. The top left figure is the original image with initial contour and the top right figure is the initial segmented image. The 2nd row gives the results after one cycle and

[1] Here cycle means outer loop.

[2] If ϕ^k and ϕ^{k+1} are values of ϕ in two consecutive cycles then the relative residual is rr $= \dfrac{\|\phi^{k+1} - \phi^k\|_2}{\|\phi^k\|_2}$

3rd row shows the final results after 3 cycles with relative residual $\mathbf{rr} = 2.016 \times 10^{-15}$. In Fig. 4.3 our method is tested on a real cameraman image and is successfully segmented. The top left figure is the original image with initial contour and top left figure is the initial segmented image. 2nd row gives results after one cycle and 3rd row gives final results after 5 cycles to get relative residual $\mathbf{rr} = 4.457 \times 10^{-15}$. In Fig. 4.4 and Fig. 4.5 our MG algorithm is tested on MRI images. In Fig. 4.6 and Fig. 4.7 MG algorithm 8 is tested on blurred images and are segmented successfully.

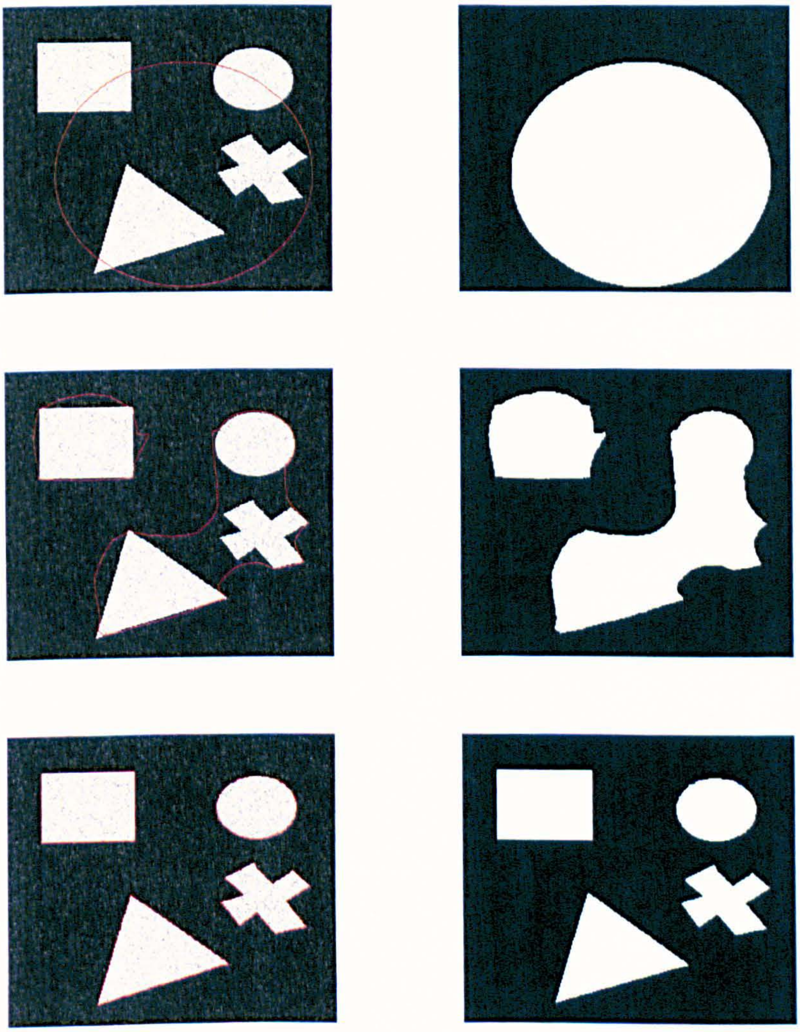


Figure 4.1: MG method with local smoother I for CV model, tested on an synthetic noisy image, $\mu = 0.1 \times (256)^2, \mathbf{rr} = 1.246 \times 10^{-15}, \beta = 1 \times 10^{-6}, \epsilon = 1 \times 10^{-20}, \phi_0 = \sqrt{(x-150)^2 + (y-140)^2} - 100$. Top left: Original image with initial contour. Top right: Initial segmented image. 2nd row: Results after 1 cycle. 3rd row: Final results in 3 cycles.

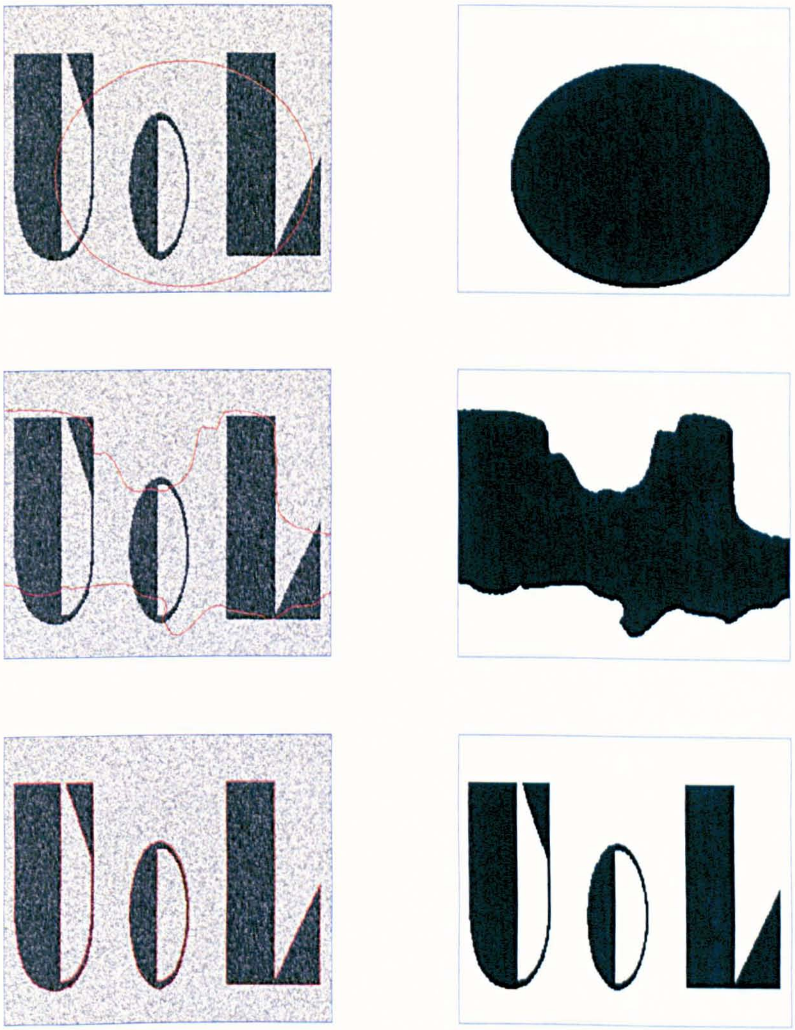


Figure 4.2: MG method with local smoother I for CV model, tested on an artificial noisy image, to see if the inner boundary is detected. Parameters used are $\mu = 0.1 \times (256)^2$, $\mathbf{rr} = 2.016 \times 10^{-15}, \beta = 1 \times 10^{-6}, \epsilon = 1 \times 10^{-20}, \phi_0 = \sqrt{(x-150)^2 + (y-140)^2} - 100$. Top left: Original image with initial contour. Top right: Initial segmented image. 2nd row: Results after 1 cycle. 3rd row: Final results in 3 cycles.

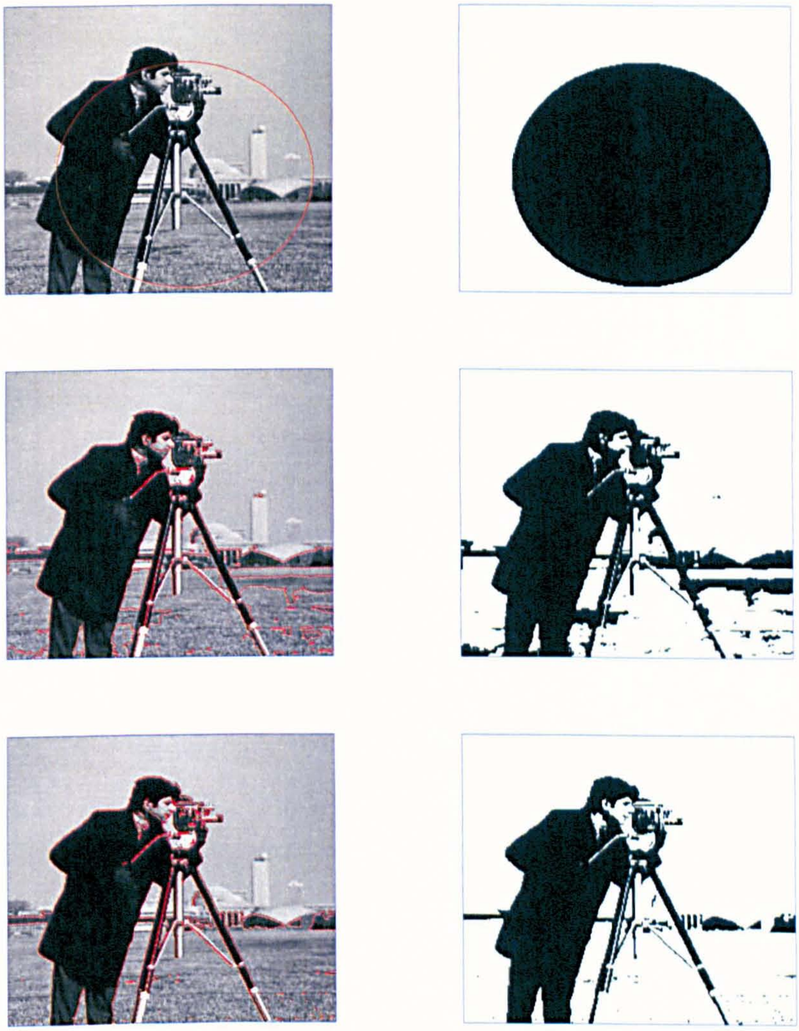


Figure 4.3: MG method with local smoother I for CV model, tested on a real image of cameraman, parameters used are $\mu = 0.0331 \times (256)^2$, $\mathbf{rr} = 4.457 \times 10^{-15}$, $\beta = 1 \times 10^{-6}$, $\epsilon = 1 \times 10^{-20}$, $\phi_0 = \sqrt{(x-150)^2 + (y-140)^2} - 100$. Top left: Original image with initial contour. Top right: Initial segmented image. 2nd row: Results after 1 cycle. 3rd row: Final results in 5 cycles.

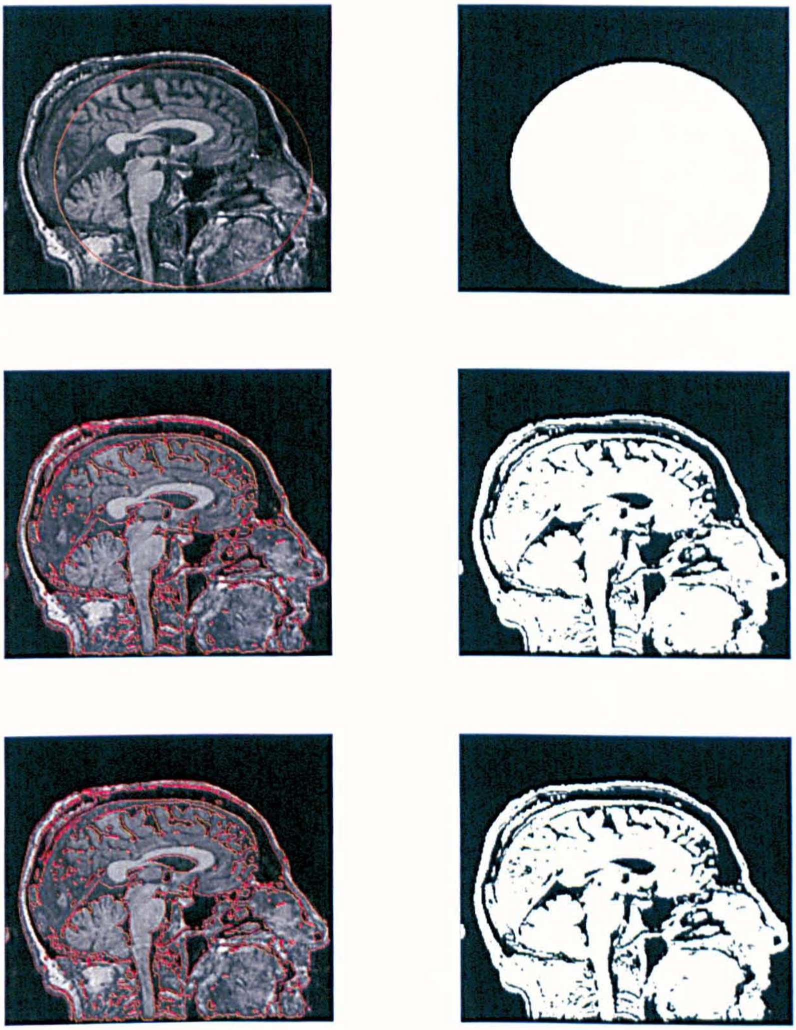


Figure 4.4: MG method with local smoother I for CV model, tested on a real MRI image, parameters used are $\mu = 0.0011 \times (256)^2, \mathbf{rr} = 3.649 \times 10^{-14}, \beta = 1 \times 10^{-6}, \epsilon = 1 \times 10^{-20}, \phi_0 = \sqrt{(x-150)^2 + (y-140)^2} - 100$. Top left: Original image with initial contour. Top right: Initial segmented image. 2nd row: Results after 1 cycle. 3rd row: Final results in 5 cycles.

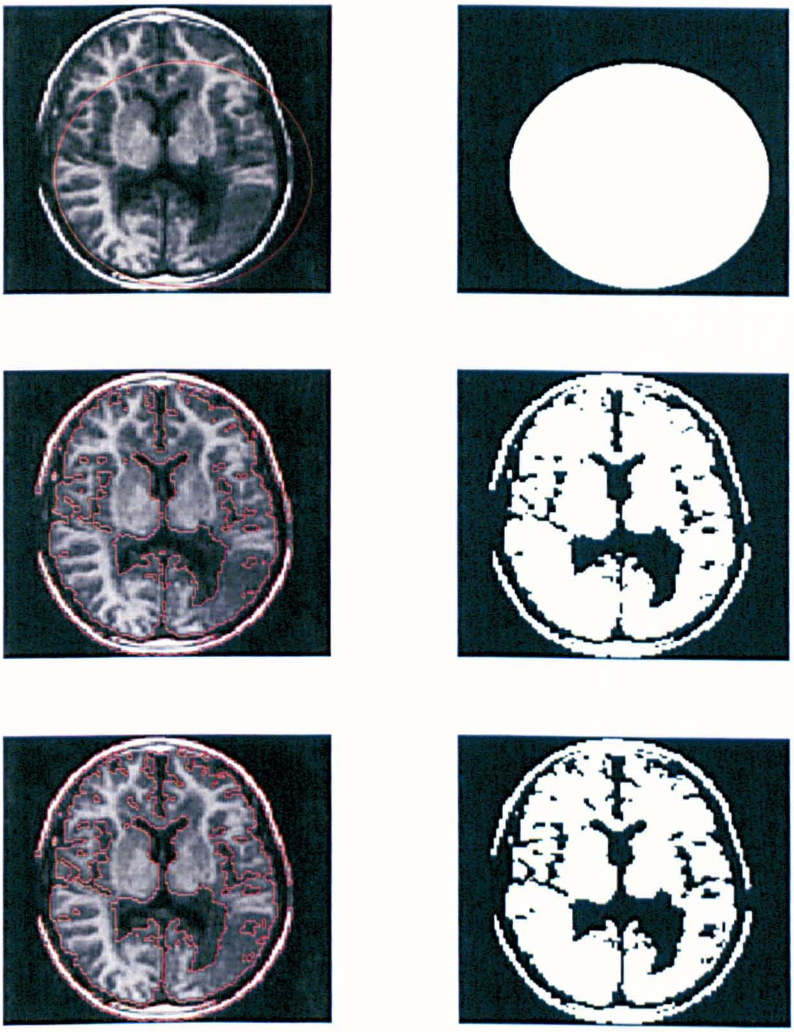


Figure 4.5: MG method with local smoother I for CV model, tested on a real MRI image, parameters used are $\mu = 0.001 \times (256)^2, \mathbf{rr} = 4.329 \times 10^{-14}, \beta = 1 \times 10^{-6}, \epsilon = 1 \times 10^{-20}, \phi_0 = \sqrt{(x-150)^2 + (y-140)^2} - 100$. Top left: Original image with initial contour. Top right: Initial segmented image. 2nd row: Results after 1 cycle. 3rd row: Final results in 5 cycles.

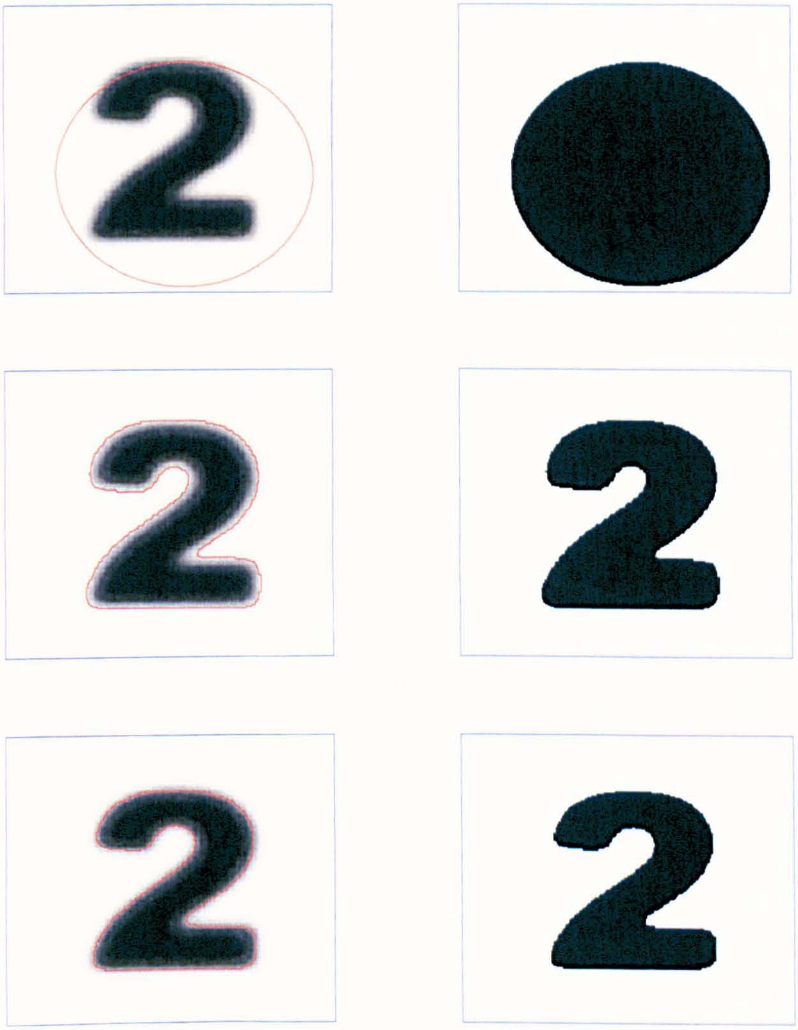


Figure 4.6: MG method with local smoother I for CV model, tested on an blurred image, parameters used are $\mu = 0.0001 \times (256)^2, \mathbf{rr} = 1.78 \times 10^{-12}, \beta = 1 \times 10^{-6}, \epsilon = 1 \times 10^{-20}, \phi_0 = \sqrt{(x-150)^2 + (y-140)^2} - 100$. Top left: Original image with initial contour. Top right: Initial segmented image. 2nd row: Results after 1 cycle. 3rd row: Final results in 6 cycles.

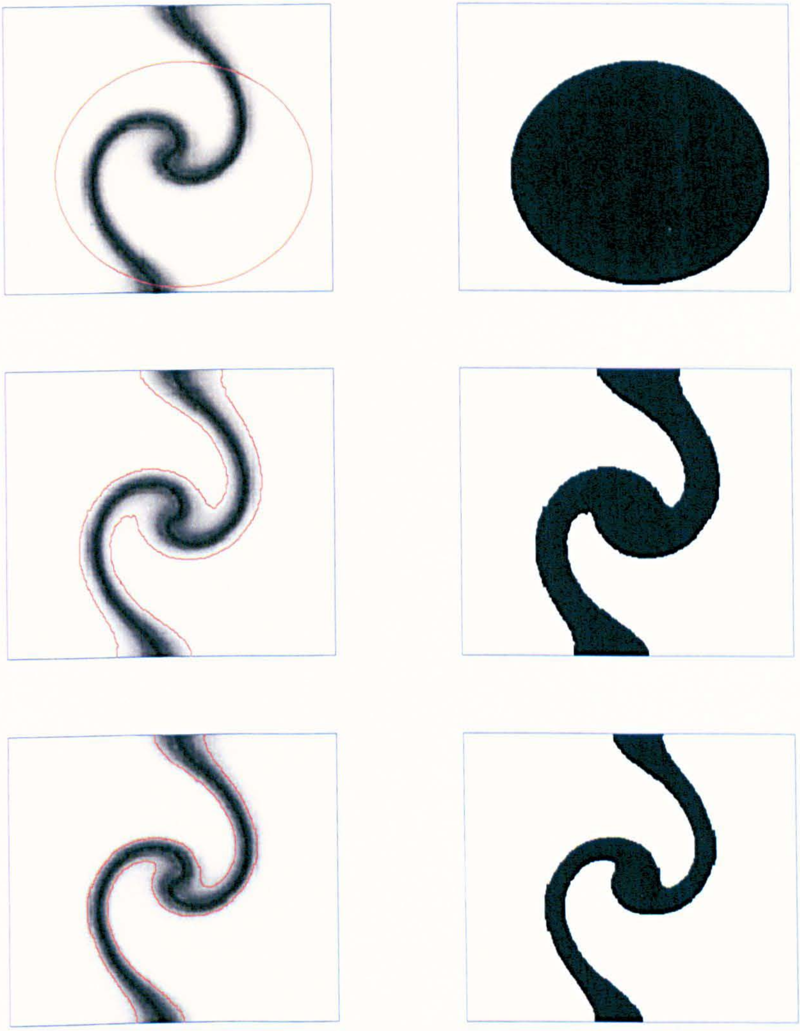


Figure 4.7: MG method with local smoother I for CV model, tested on an blurred image, parameters used are $\mu = 0.0001 \times (256)^2, \mathbf{rr} = 1.101 \times 10^{-12}, \beta = 1 \times 10^{-6}, \epsilon = 1 \times 10^{-20}, \phi_0 = \sqrt{(x-150)^2 + (y-140)^2} - 100$. Top left: Original image with initial contour. Top right: Initial segmented image. 2nd row: Results after 1 cycle. 3rd row: Final results in 6 cycles.

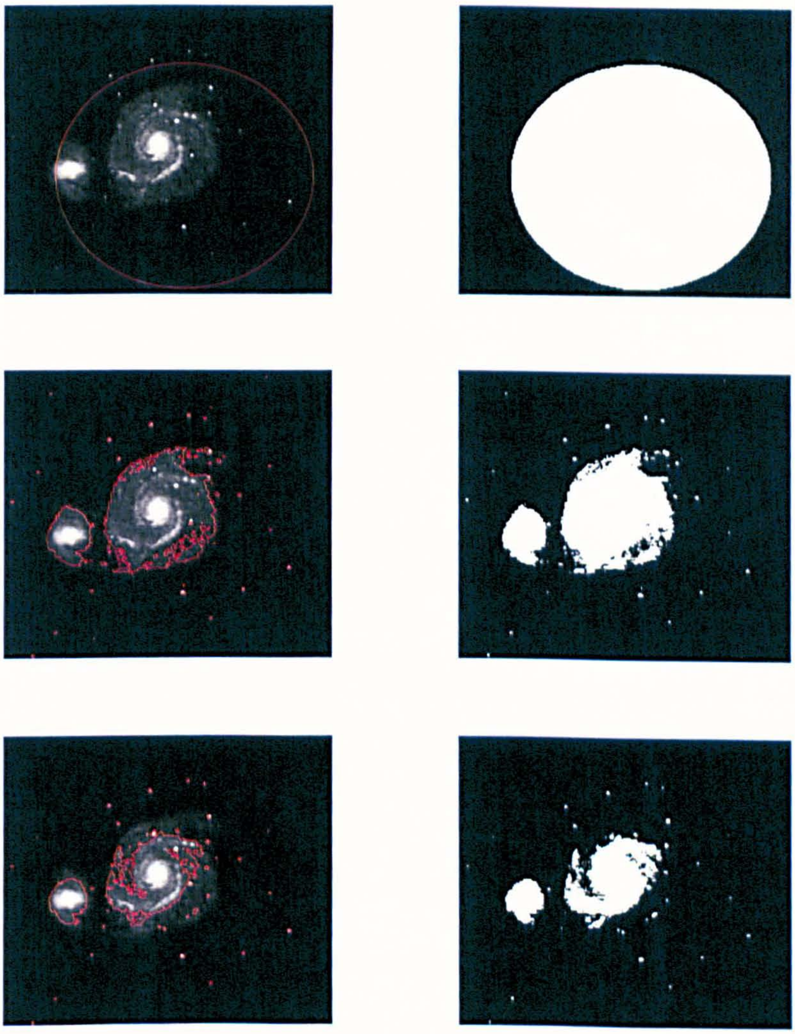


Figure 4.8: MG method with local smoother I for CV model, tested on an galaxy image, parameters used are $\mu = 0.0001 \times (256)^2, \mathbf{rr} = 3.05 \times 10^{-14}, \beta = 1 \times 10^{-6}, \epsilon = 1 \times 10^{-20}, \phi_0 = \sqrt{(x-150)^2 + (y-140)^2} - 100$. Top left: Original image with initial contour. Top right: Initial segmented image. 2nd row: Results after 2 cycles. 3rd row: Final results in 10 cycles.

4.5.2 Results using smoother II

In this section we test MG algorithm 8 with smoother II (global smoother). In Fig. 4.9, MG algorithm 8 with smoother II is tested on an synthetic noisy image. Parameters used are the same as used Fig. 4.1. The final results are given with relative residual $\mathbf{rr} = 1.246 \times 10^{-15}$ in 4 cycles.

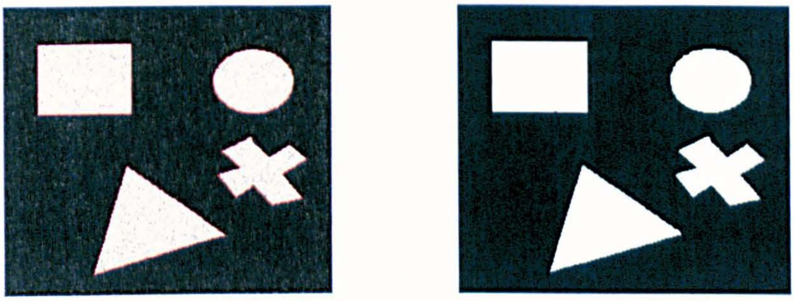


Figure 4.9: MG algorithm with smoother II for CV model, tested on a synthetic noisy image with $\mu = 0.1 \times (256)^2, \mathbf{rr} = 1.246 \times 10^{-15}$ obtained with 4 cycles.

4.5.3 Improved solution for global minimizers

Here we give some evidence of obtaining improved solution of the global minimizer. As in [15], X. Bresson et al discussed the drawbacks of the variational segmentation model, with the main one being the existence of local minima in the energy. In non-convex optimization, the local minima often lead to unsatisfactory results. In Fig. 4.10, 4.11 our MG algorithm on the problems addressed in [25, 15, 41], which are segmented correctly with this particular initial guess ϕ_0. On the other hand, in Fig. 4.12 we have displayed the results obtained by using AOS method, where the method gets stuck in a local minima with this particular initial guess given in Fig. 4.10, 4.11. The same sort of results are found using semi implicit (SI) method. We have tested this for many problems and see the same type of difference with the other methods like AOS and SI.

4.5.4 Convergence tests and full multigrid

Our preliminary results suggest that our MG (with smoother I) can converge for varying choices of the initial ϕ e.g specifying ϕ having a small circle at a fixed position. We have tried different initial guesses for instance a rectangle, a parallelogram etc., all of them work. For simple artificial images only 3-4 MG cycles to get the relative residual below 10^{-12} and for real images like MRI images we need 7-8 MG cycles to get relative residuals less than 10^{-12}.

However to eliminate the need of an initial guess, we consider the use of a full multigrid (FMG) idea [118] which starts the solution of Eq. (4.9) on the coarsest grid. Then each solution is interpolated onto the next fine grid to give an initial guess until we reach the finest grid where we start the MG algorithm.

To test on the scalability of MG, we display in Table 4.2 the number of steps needed to reach a desirable accuracy for the images 1 and 7 from Fig. 3.1. The CPU times (in seconds) are obtained from running Matlab on a Pentium PC for illustration purpose. Here FMG means that the initial guess of ϕ is obtained from a FMG method following

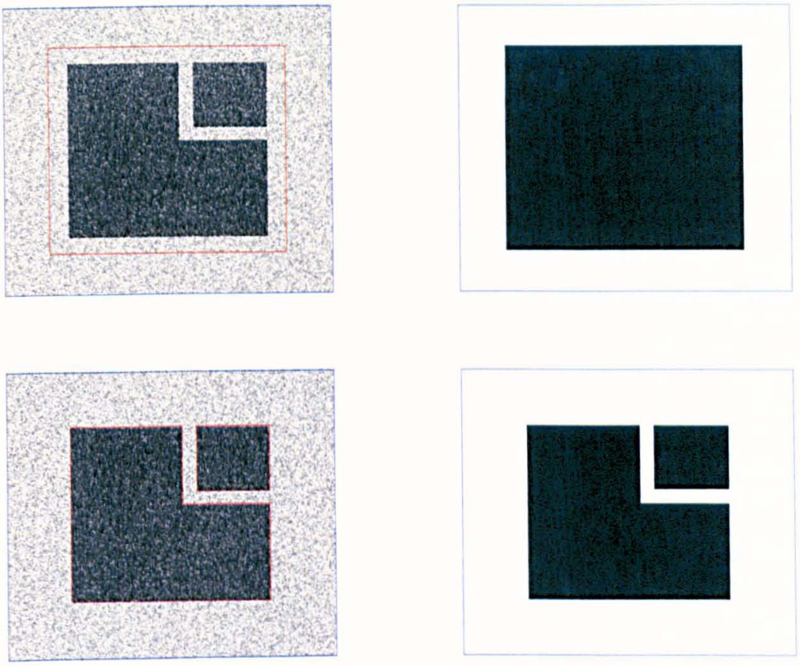


Figure 4.10: Improved solution towards global minimizer, tested on box image using MG algorithm 8 in 3 cycles. Top Row: Initial data. Bottom Row: Final results.

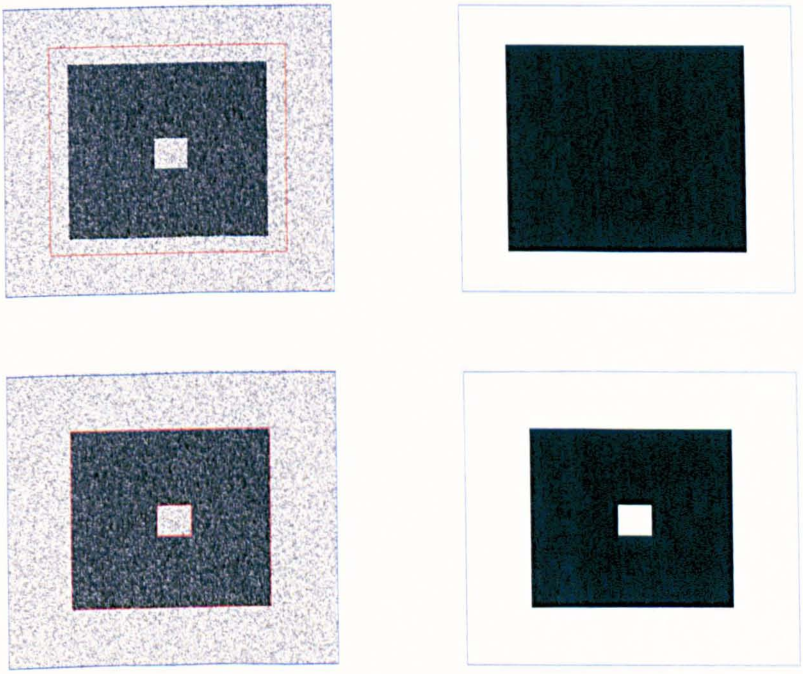


Figure 4.11: Another example for improved solution towards global minimizer, tested on box image using MG algorithm 8 in 3 cycles. Top Row: Initial data. Bottom Row: Final results.

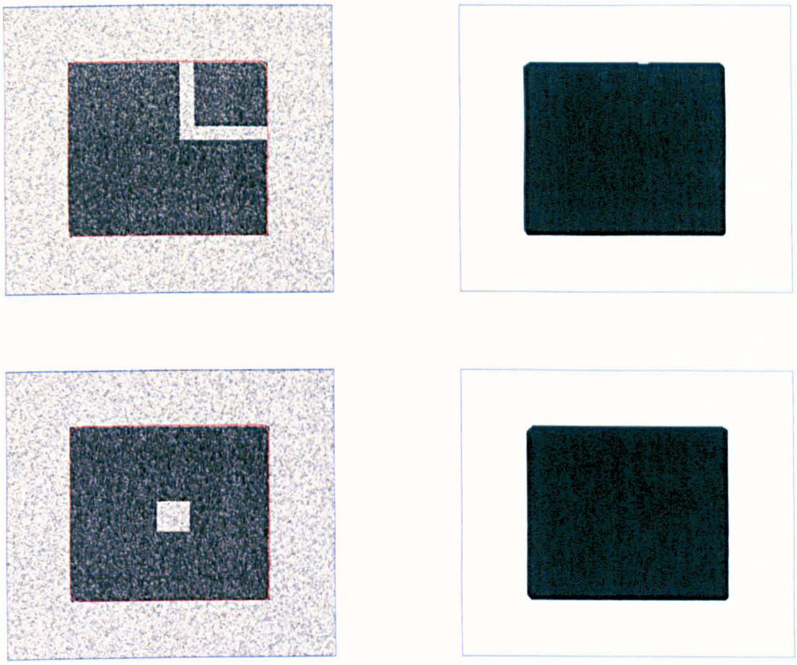


Figure 4.12: These are the results obtained by using AOS method which is stuck at a local minimum i.e could not segment the images.

by the normal MG cycles. Clearly MG (with smoother I) is efficient. Our MG works the best with $\nu_1 = \nu_2 = 2$, but it will not work well if much less smoothing steps are used e.g. $\nu_1 = 1, \nu_2 = 0$.

Table 4.2: Test of scalability of MG and FMG. CAM: Cameraman image and UOL: UOL image given in Fig 3.1

Image	Method	Image size:	MG cycles	CPU (seconds)
CAM	MG	128^2	4	9
		256^2	5	15
		512^2	5	27
		1024^2	5	90
	FMG	128^2	3	15
		256^2	3	17
		512^2	3	27
		1024^2	3	83
UOL	MG	128^2	2	5
		256^2	2	6
		512^2	2	12
		1024^2	2	50
	FMG	128^2	2	11
		256^2	2	13
		512^2	2	21
		1024^2	2	61

4.5.5 Comparison of complexity and CPU saving

Finally in this chapter we compare the speed of MG (with both smoothers) with semi implicit (SI) method and additive operator splitting (AOS) methods. We first estimate the computational complexity of the algorithms involved. To be concrete, we assume that the inner solver of a SI method is by a conjugate gradient method for 25 steps, and in the MG method $\nu_1 = 2$ and $\nu_2 = 1$. Then, consider segmenting some image of size $m \times m$. Then setup cost for the 4 main coefficients is about $4 \times 8m^2 = 32m^2$ operations for all methods per step. The cost of each step of SI method is thus $W_1 = 25 \times 5m^2 + 32m^2 = 157m^2$. For the AOS, solving each tridiagonal matrix costs $4m^2$ operations so the cost of each step is $W_2 = 2 \times 4m^2 + 32m^2 = 40m^2$. Finally the finest level smoothing cost for MG is $11m^2 \times (\nu_1 + \nu_2) = 33m^2$ so the finest level cost is $33m^2 + 32m^2 = 65m^2$. The cost per MG over all levels is $W_3 = 4/3 \times 65m^2 \approx 87m^2$. Therefore the practical efficiency of these methods, although all of $\mathcal{O}(N) = \mathcal{O}(m \times m)$ complexity per step, will depend on the number of actual iterations steps used for achieving the same accuracy.

In Table 4.3 we have presented the comparison of our method with SI and AOS methods. The terms used in the heading of Table 4.3 have the following meanings:

Size: The size of given image $m \times n$.

Itr: Number of iterations used to get the required result.

CPU(s): Time in seconds required for CPU to perform these iterations.

SI: Semi Implicit method.

AOS: Additive Operator Splitting

MG: Multigrid Method.

ART: Artificial image, problem 7 in Fig. 3.1 and **REAL:** Real image like MRI, problem 5 in Fig. 3.1.

– : Results with high CPU or out of memory.

S-I: Smoother I.

S-II: Smoother II.

AOS multi-resolution: AOS method is implemented in coarse to fine level manner, i.e AOS method is used to solve the problem on coarsest level and interpolate the solution to the fine level and use it as initial guess, to solve the problem on fine level using AOS method and so on until the finest level is reached.

From Table 4.3 we see that the MG method is as fast as the SI method and AOS method for images of small sizes, but it is more efficient for the large images, where the above mentioned methods are very slow or not working.

Table 4.3: Comparison of MG with SI and AOS methods

Prob.	Size	AOS method		AOS multi-resolution		SI Method		MG Method (S-I)		MG Method (S-II)	
		Itr	CPU(s)	Itr	CPU(s)	Itr	CPU(s)	Itr	CPU(s)	Itr	CPU(s)
ART	128^2	60	4.8	60	4.8	80	16.5	2	8.5	2	8
	256^2	140	50	80	34	100	90.3	2	9.4	3	13.4
	512^2	280	421	170	277	439	1.3×10^4	2	13	3	17
	1024^2	1200	7661	240	1630	–	–	2	27	3	32
	2048^2	–	–	–	–	–	–	2	90	3	100
REAL	128^2	100	10.5	100	10.5	130	32.2	3	12.8	4	15
	256^2	280	110.5	156	68	180	450	3	14	4	22.2
	512^2	800	1230	312	503	–	$- - \times 10^4$	3	19.2	4	22.2
	1024^2	–	–	–	–	–	–	3	40.7	4	42
	2048^2	–	–	–	–	–	–	3	133	4	136.9

4.6 Conclusions

We have proposed an effective smoother for a nonlinear multigrid method to solve the Chan-Vese active contour without edges model. A linear Fourier analysis shows that our local smoother I is better than the global smoother II. For large images, as expected, our MG leads to much faster solutions than the uni-level methods of SI and AOS. A somewhat surprising observation is that our MG can reach closer to the global minimizer than the SI and AOS methods in all of our test cases.

Chapter 5

The Multigrid Algorithms for Variational Multiphase Image Segmentation

The main theme of this chapter is to develop fast iterative methods for solving multiphase image segmentation models. Firstly we extend the 2 phase image segmentation CV model [35] discussed in chapter 4 to a multiphase image segmentation model [120] and then extend the multigrid algorithm proposed in chapter 4 towards multiphase image segmentation.
The main references for this chapter are [8, 14, 23, 27, 35, 72, 74, 120, 127].

5.1 Introduction

Segmentation, referring to separating image features from backgrounds, is one of the most important tasks arising from computer vision (e.g. detecting objects) and many image processing fields (e.g. picking out special cells in cell imaging). Segmentation methods fall into several categories, including histogram analysis, region growing, edge detection and PDE-based variational methods. Our main focus will be on nonlinear PDE based segmentation methods.

The PDE and variational based methods [87] are among the more recently developed tools for image segmentation. The snake model [75], the active contours model [21], the gradient vector flow method [129] and the curvature driven diffusion method [33] all belong to this class of PDE-based methods. In most situations, the level set method described in [90] proves to be an indispensable tool for analysis and implementation.

Our primary aim in this chapter is an extension of the Badshah-Chen multigrid method [8] (already discussed in chapter 4) to the Vese-Chan multiphase image segmentation [120]. Two problems are encountered in this work. Firstly a local Fourier analysis (LFA) of the generalized local smoother suggests that it is not effective if a small number (e.g. $2-3$) of smoothing steps is used. A closer study shows that this ineffectiveness is due to a few image pixels only, where the linearized coefficients differ greatly. We then propose a different smoothing (under-relaxation) strategy at these 'odd' pixels. The LFA shows that the modified smoother is effective. Secondly we found that the Vese-Chan multiphase model [120] may not segment images (i.e. may not converge to the desired level set functions) if good initial guesses are not provided; this problem is inherent in the model rather than the numerical solution methods. To overcome this latter problem we adopt the idea described in [72] to decouple the multiphase model into repeated two phase models in a hierarchical way. This prompts us to consider a multigrid method for each decoupled two phase problem. It turns out that this hierarchical approach leads to a fast algorithm that can segment an image for any (tested) initial guesses (including previously failed cases of initial guesses). Moreover, even when good initial guesses are available, the hierarchical approach is also faster (in number of iterations and CPU) than the multiphase multigrid method without the modified smoother; with the modified smoother, the two algorithms are comparable in speed. In Section 5.2, we describe the Vese-Chan multiphase model. In Section 5.3, we develop the multigrid method for solving partial differential equation arisen from the minimization of the multiphase model. We also give the Fourier analysis of the smoother. In Section 5.4, we implement the 2-phase image segmentation model in hierarchical way to get multiphase segmentation results. We end of the chapter with some numerical experiments and conclusion.

5.2 Multigrid algorithm I for multiphase segmentation (MG1)

The Vese-Chan multiphase segmentation model [120] is the extension of the 2-phase Chan-Vese segmentation model [35] which has already been discussed in chapter 4. As described in [120], with one level set function, we can segment an image into two phases as one level set cannot represent more than two phases. In general, to divide an image into n phases, we need $\log_2 n$ level set functions. We remark that related work by Tai et al [78] and Ambrosio-Tortorelli [4] provides alternatives to these multiple level set functions.

Consider $p = \log_2 n$ level set functions $\phi_\ell : \Omega \to \mathbb{R}$ for $\ell = 1, 2, \ldots, p$. The union of the zero level sets of all ϕ_ℓ will represent the edges in the segmented image. For $1 \leqslant s \leqslant n = 2^p$, denote by $c_s = \text{mean}(z)$ the average value of image grey-scales in phase s and by χ_s the characteristic function for phase s. Then the proposed minimization energy for multiphase segmentation by Vese-Chan [120] is the following:

$$F_n(c, \Phi) = \sum_{1 \leqslant s \leqslant n} \int_\Omega (z(x,y) - c_s)^2 \chi_s dxdy \; + \; \mu \sum_{1 \leqslant \ell \leqslant p} \int_\Omega |\nabla H(\phi_\ell)| dxdy \qquad (5.1)$$

where $c = (c_1, c_2, \ldots, c_n)$ and $\Phi = (\phi_1, \phi_2, \ldots, \phi_p)$; note $n = 2^p$. In this chapter we mainly focus on the 4-phase segmentation i.e. $n = 4$ or $p = 2$, which we denote by SEG4. But all the ideas will carry over to more phases.

We shall consider the following minimization problem:

$$\min_{c, \Phi} F_4(c, \Phi), \qquad (5.2)$$

where

$$\begin{aligned} F_4(c,\Phi) \;=\; & \int_\Omega (z(x,y) - c_{11})^2 H(\phi_1) H(\phi_2) dxdy + \int_\Omega (z(x,y) - c_{10})^2 H(\phi_1)(1 - H(\phi_2)) dxdy \\ & + \int_\Omega (z(x,y) - c_{01})^2 (1 - H(\phi_1)) H(\phi_2) dxdy + \; \mu \int_\Omega |\nabla H(\phi_1)| dxdy \\ & + \int_\Omega (z(x,y) - c_{00})^2 (1 - H(\phi_1))(1 - H(\phi_2)) dxdy + \mu \int_\Omega |\nabla H(\phi_2)| dxdy \qquad (5.3) \end{aligned}$$

where $c = (c_{11}, c_{10}, c_{01}, c_{00})$ and $\Phi = (\phi_1, \phi_2)$. Here the phase domains will be interlaced by the zero level sets of ϕ_1, ϕ_2 i.e. $\Omega_1 = \{(x,y) \; : \; \phi_1 > 0, \phi_2 > 0\}$, $\Omega_2 = \{(x,y) \; : \; \phi_1 > 0, \phi_2 < 0\}$. Once Φ is found, the segmented image u is

$$u = c_{11} H(\phi_1) H(\phi_2) + c_{10} H(\phi_1)(1 - H(\phi_2)) + c_{01}(1 - H(\phi_1)) H(\phi_2) + c_{00}(1 - H(\phi_1))(1 - H(\phi_2)).$$

Minimizing (5.2) with respect to c and Φ, we have:

$$c_{11}(\phi) = \frac{\int_\Omega zH(\phi_1)H(\phi_2)dxdy}{\int_\Omega H(\phi_1)H(\phi_2)dxdy}, \qquad c_{10}(\phi) = \frac{\int_\Omega zH(\phi_1)(1-H(\phi_2))dxdy}{\int_\Omega H(\phi_1)(1-H(\phi_2))dxdy},$$
$$c_{01}(\phi) = \frac{\int_\Omega z(1-H(\phi_1))H(\phi_2)dxdy}{\int_\Omega (1-H(\phi_1))H(\phi_2)dxdy}, \qquad c_{00}(\phi) = \frac{\int_\Omega z(1-H(\phi_1))(1-H(\phi_2))dxdy}{\int_\Omega (1-H(\phi_1))(1-H(\phi_2))dxdy}$$

and the following Euler-Lagrange's equations

$$\begin{cases} \delta_\epsilon(\phi_1)\Big[\mu\nabla\cdot\frac{\nabla\phi_1}{|\nabla\phi_1|} - [T_1H_\epsilon(\phi_2) + T_2(1-H_\epsilon(\phi_2))]\Big] = 0, \\ \delta_\epsilon(\phi_2)\Big[\mu\nabla\cdot\frac{\nabla\phi_2}{|\nabla\phi_2|} - [T_1H_\epsilon(\phi_1) + T_2(1-H_\epsilon(\phi_1))]\Big] = 0, \end{cases} \tag{5.4}$$

with Neumann boundary conditions, where $T_1 = (z-c_{11})^2 - (z-c_{01})^2$ and $T_2 = (z - c_{10})^2 - (z-c_{00})^2$. We shall shortly discuss how to solve (5.4) efficiently.

An easy but less efficient alternative is to solve the following evolution problem

$$\begin{cases} \frac{\partial\phi_1}{\partial t} = \delta_\epsilon(\phi_1)\Big[\mu\nabla\cdot\frac{\nabla\phi_1}{|\nabla\phi_1|} - [T_1H_\epsilon(\phi_2) + T_2(1-H_\epsilon(\phi_2))]\Big], \\ \frac{\partial\phi_2}{\partial t} = \delta_\epsilon(\phi_2)\Big[\mu\nabla\cdot\frac{\nabla\phi_2}{|\nabla\phi_2|} - [T_1H_\epsilon(\phi_1) + T_2(1-H_\epsilon(\phi_1))]\Big], \end{cases} \tag{5.5}$$

with initial conditions $\phi_1(0,x,y) = \phi_{1,0}(x,y), \phi_2(0,x,y) = \phi_{2,0}(x,y)$. In [120] these parabolic equations were solved using the additive operator-splitting (AOS, semi-implicit) method (details are given in chapter 4) which will be used later for comparison.

5.3 Multigrid algorithm I

Instead of solving (5.5), we consider solving (5.4). Let $(\phi_\ell)_{i,j} = \phi_\ell(x_i, y_j)$, for $\ell = 1, 2$. Using finite differences schemes to discretize (5.4) for ϕ_ℓ, the equations at a pixel point (i,j) are given by

$$\begin{cases} \delta_\epsilon(\phi_1)_{i,j}\Big\{\frac{\Delta^x_-}{h_1}\frac{\mu\Delta^x_+(\phi_1)_{i,j}/h_1}{\sqrt{(\Delta^x_+(\phi_1)_{i,j}/h_1)^2 + (\Delta^y_+(\phi_1)_{i,j}/h_2)^2 + \beta}} - (T_1)_{i,j}H_\epsilon(\phi_2)_{i,j} + \\ \frac{\Delta^y_-}{h_2}\frac{\mu\Delta^y_+(\phi_1)_{i,j}/h_2}{\sqrt{(\Delta^x_+(\phi_1)_{i,j}/h_1)^2 + (\Delta^y_+(\phi_1)_{i,j}/h_2)^2 + \beta}} - (T_2)_{i,j}(1 - H_\epsilon(\phi_2)_{i,j})\Big\} = 0, \\ \delta_\epsilon(\phi_2)_{i,j}\Big\{\frac{\Delta^x_-}{h_1}\frac{\mu\Delta^x_+(\phi_2)_{i,j}/h_1}{\sqrt{(\Delta^x_+(\phi_2)_{i,j}/h_1)^2 + (\Delta^y_+(\phi_2)_{i,j}/h_2)^2 + \beta}} - (T_1)_{i,j}H_\epsilon(\phi_1)_{i,j} + \\ \frac{\Delta^y_-}{h_2}\frac{\mu\Delta^y_+(\phi_2)_{i,j}/h_2}{\sqrt{(\Delta^x_+(\phi_2)_{i,j}/h_1)^2 + (\Delta^y_+(\phi_2)_{i,j}/h_2)^2 + \beta}} - (T_2)_{i,j}(1 - H_\epsilon(\phi_1)_{i,j})\Big\} = 0, \end{cases} \tag{5.6}$$

where $(T_1)_{i,j} = (z_{i,j} - c_{11})^2 - (z_{i,j} - c_{01})^2$ and $(T_2)_{i,j} = (z_{i,j} - c_{10})^2 - (z_{i,j} - c_{00})^2$.

Let $\underline{\mu} = \mu/h_1$, $\underline{\beta} = h_1^2\beta$ and $\lambda = h_1/h_2$. Also denote $(f_1)_{i,j} = (T_1)_{i,j}H_\epsilon(\phi_2)_{i,j} + T_2)_{i,j}(1 - H_\epsilon(\phi_2)_{i,j})$ and $(f_2)_{i,j} = (T_1)_{i,j}H_\epsilon(\phi_1)_{i,j} + T_2)_{i,j}(1 - H_\epsilon(\phi_1)_{i,j})$. For given ϕ_1 and ϕ_2 we compute f_1 and f_2, keeping f_1 and f_2 fixed and we solve coupled PDEs in equation (5.6) for ϕ_1 and ϕ_2 using multigrid method. Once ϕ_1 and ϕ_2 are found we update f_1 and

f_2 and so on. For $\ell = 1, 2$, denote the coefficients (to be frozen) by

$$D_\ell(\phi_\ell)_{i,j} = \frac{1}{\sqrt{(\Delta_+^x(\phi_\ell)_{i,j})^2 + (\lambda\Delta_+^y(\phi_\ell)_{i,j})^2 + \bar\beta}},$$

$$D_\ell(\phi_\ell)_{i-1,j} = \frac{1}{\sqrt{(\Delta_+^x(\phi_\ell)_{i-1,j})^2 + (\lambda\Delta_+^y(\phi_\ell)_{i-1,j})^2 + \bar\beta}},$$

$$D_\ell(\phi_\ell)_{i,j-1} = \frac{1}{\sqrt{(\Delta_+^x(\phi_\ell)_{i,j-1})^2 + (\lambda\Delta_+^y(\phi_\ell)_{i,j-1})^2 + \bar\beta}}.$$

We can simplify equation (5.6) to

$$\begin{aligned}&\Big[D_\ell(\phi_\ell)_{i,j}((\phi_\ell)_{i+1,j} - (\phi_\ell)_{i,j}) - D_\ell(\phi_\ell)_{i-1,j}((\phi_\ell)_{i,j} - (\phi_\ell)_{i-1,j})\Big] \\ &+\lambda^2\Big[D_\ell(\phi_\ell)_{i,j}((\phi_\ell)_{i,j+1} - (\phi_\ell)_{i,j}) - D_\ell(\phi_\ell)_{i,j-1}((\phi_\ell)_{i,j} - (\phi_\ell)_{i,j-1})\Big] = (\bar f_\ell)_{i,j},\end{aligned} \tag{5.7}$$

where $\bar f_\ell = f_\ell/\mu$, with the boundary conditions

$$(\phi_\ell)_{i,0} = (\phi_\ell)_{i,1}, \quad (\phi_\ell)_{i,m_2+1} = (\phi_\ell)_{i,m_2}, \quad (\phi_\ell)_{0,j} = (\phi_\ell)_{1,j}, \quad (\phi_\ell)_{m_1+1,j} = (\phi_\ell)_{m_1,j}. \tag{5.8}$$

Let $\widetilde{\phi_\ell}$ be the approximation to ϕ_ℓ at the current iteration. Then from equation (5.7), pursuing only local unknowns ϕ_ℓ at (i, j), we have the following linear equations

$$\begin{aligned}&\Big[D_\ell(\widetilde{\phi_\ell})_{i,j}((\widetilde{\phi_\ell})_{i+1,j} - (\phi_\ell)_{i,j}) - D_\ell(\widetilde{\phi_\ell})_{i-1,j}((\phi_\ell)_{i,j} - (\widetilde{\phi_\ell})_{i-1,j})\Big] \\ &+\lambda^2\Big[D_\ell(\widetilde{\phi_\ell})_{i,j}((\widetilde{\phi_\ell})_{i,j+1} - (\phi_\ell)_{i,j}) - D_\ell(\widetilde{\phi_\ell})_{i,j-1}((\phi_\ell)_{i,j} - (\widetilde{\phi_\ell})_{i,j-1})\Big] = (\bar f_\ell)_{i,j}.\end{aligned} \tag{5.9}$$

Our proposed algorithm solves these equations for $(\phi_\ell)_{i,j}$ to update $(\widetilde{\phi_\ell})_{i,j}$ which leads to updating the coefficients locally and further iterations (before moving to the next pixel in a Gauss-Seidel fashion). Denote the system of non-linear equations from (5.7) by

$$\begin{cases} N_1^h(\phi_1^h) = \bar f_1^h, \\ N_2^h(\phi_2^h) = \bar f_2^h, \end{cases} \tag{5.10}$$

where ϕ_ℓ^h and $\bar f_\ell^h$ are grid functions on an $m_1 \times m_2$ cell centered rectangular grid Ω^h with spacing $h = (h_1, h_2)$ and $l = 1, 2$.

We shall first summarize this local smoother and then present our algorithm I.

Algorithm 9 (Smoother for multiphase model) *Let a smoothing step for (5.7) via (5.9) be*

$$\phi_\ell^h \longleftarrow Smoother(\phi_\ell^h, \bar{f_\ell}^h, maxit, tol)$$

where $\ell = 1, 2$ and h is the step size on level Ω^h.

```
for $i = 1 : m_1$
    for $j = 1 : m_2$
        for iter=1:maxit
        $\widetilde{\phi_\ell}^h \leftarrow \phi_\ell^h$,   $A_\ell = D_\ell(\widetilde{\phi_\ell})_{i,j}^h(\widetilde{\phi_\ell})_{i+1,j}^h + D_\ell(\widetilde{\phi_\ell})_{i-1,j}^h(\widetilde{\phi_\ell})_{i-1,j}^h$,
                  $B_\ell = D_\ell(\widetilde{\phi_\ell})_{i,j}^h(\widetilde{\phi_\ell})_{i,j+1}^h + D_\ell(\widetilde{\phi_\ell})_{i,j-1}^h(\widetilde{\phi_\ell})_{i,j-1}^h$,

        $(\phi_\ell)_{i,j}^h = \dfrac{A_\ell + \lambda^2 B_\ell - \bar f_{\ell i,j}}{D_\ell(\widetilde{\phi_\ell})_{i,j}^h + D_\ell(\widetilde{\phi_\ell})_{i-1,j}^h + \lambda^2(D_\ell(\widetilde{\phi_\ell})_{i,j}^h + D_\ell(\widetilde{\phi_\ell})_{i,j-1}^h)}$
        if $|(\phi_\ell)_{i,j}^h - (\tilde{\phi}_\ell)_{i,j}^h| < tol$  Stop
        end
    end
end
```

Multigrid Algorithm: Equation (5.10) will be solved by the following multigrid algorithm [8, 14, 39, 118]. This is the full approximation scheme of Brandt [14].

Algorithm 10 (Multigrid Algorithm) *Assume we have set up these multigrid parameters:*

ν_1 *pre-smoothing steps on each level*

ν_2 *post-smoothing steps on each level*

γ *the number of multigrid cycles on each level ($\gamma = 1$ for V-cycling and $\gamma = 2$ for W-cycling). Set tol $= 0.1$. Here we present one step V-cycle of the nonlinear multigrid method for SEG4. First for $\ell = 1, 2$ compute f_ℓ^h using given ϕ_ℓ^h. A multigrid cycle refers to one call to the procedure*

Start

$$\phi_\ell^h \longleftarrow FASCYC^h(\phi_\ell^h, \bar{f_\ell}^h, \nu_1, \nu_2, tol)$$

1. *If Ω^h is the coarsest grid, then solve equation (5.10) using a time marching technique (the AOS method) and then stop.*
   Else do the Pre-Smoothing step:

$$\phi_\ell^h \longleftarrow Smoother(\phi_\ell^h, \bar{f_\ell}^h, \nu_1, tol)$$

2. **Restriction:**

$$\phi_\ell^{2h} = I_h^{2h}\phi_\ell^h, \quad \bar{\phi_\ell}^{2h} = \phi_\ell^{2h},$$
$$\bar{f_\ell}^{2h} = I_h^{2h}(\bar{f_\ell}^h - N_\ell^h\phi_\ell^h) + N_\ell^{2h}\phi_\ell^{2h},$$
$$\phi_\ell^{2h} \longleftarrow FASCYC^{2h}(\phi_\ell^{2h}, \bar{f_\ell}^{2h}, \nu_1, \nu_2)$$

3. **Interpolation**

$$\phi_\ell^h \longleftarrow \phi_\ell^h + I_{2h}^h(\phi_\ell^{2h} - \bar{\phi_\ell}^{2h})$$

4. *Implement the Post-Smoothing step:*

$$\phi_\ell^h \longleftarrow Smoother(\phi_\ell^h, \bar{f_\ell}^h, \nu_2, tol).$$

Once ϕ_ℓ^h for $\ell = 1, 2$ is found update f_ℓ.
If **rr** *< tol Stop*
Else go to **Start**.

Here the restriction operator I_h^{2h} is by full weighting and the interpolation I_{2h}^h by the bilinear operator [118].

As we know, an effective multigrid algorithm relies on two necessary ingredients: Smooth residual errors on fine levels and effective error corrections from coarse levels. In practice, the effectiveness of a smoother in smoothing errors is the key to success. In the next section we give the local Fourier analysis of our smoother in Algorithm 9.

5.3.1 Local Fourier analysis and a modified smoother

Local Fourier analysis (LFA) is a suitable tool to analyze the convergence rate of any iterative method for linear equations. However our underlying equations are nonlinear so a LFA will consider a linearized equation, and as linearization occurs locally at each pixel, we shall look for the maximum rate from all pixel locations.

Consider Algorithm 9 in the case of a square image with $m = m_1 = m_2$ and also $h_1 = h_2 = h$ for simplicity; then $\lambda = 1$. Given the previous iterate at step k, $\tilde{\phi}_\ell = \phi_\ell^{(k)}$, denote $a_1 = D_1(\widetilde{\phi_1})_{i-1,j}$, $a_2 = D_1(\widetilde{\phi_1})_{i,j}$, $a_3 = D_1(\widetilde{\phi_1})_{i,j-1}$, $b_1 = D_2(\widetilde{\phi_2})_{i-1,j}$, $b_2 = D_2(\widetilde{\phi_2})_{i,j}$, $b_3 = D_2(\widetilde{\phi_2})_{i,j-1}$ which are to be considered as local constants. From (5.7), the grid equation at (i,j) is the following

$$\left\{\begin{aligned} &-(a_1+2a_2+a_3)(\phi_1)_{i,j} + a_1(\phi_1)_{i-1,j} + a_3(\phi_1)_{i,j-1} + a_2[(\phi_1)_{i+1,j} + (\phi_1)_{i,j+1}] \\ &\qquad\qquad\qquad\qquad\qquad\qquad = (\bar{f}_1)_{i,j}, \\ &-(b_1+2b_2+b_3)(\phi_2)_{i,j} + b_1(\phi_2)_{i-1,j} + b_3(\phi_2)_{i,j-1} + b_2[(\phi_2)_{i+1,j} + (\phi_2)_{i,j+1}] \\ &\qquad\qquad\qquad\qquad\qquad\qquad = (\bar{f}_2)_{i,j}. \end{aligned}\right. \tag{5.11}$$

And our local smoother can be written as

$$\left\{\begin{aligned} &-(a_1+2a_2+a_3)(\phi_1)_{i,j}^{(k+1)} + a_1(\phi_1)_{i-1,j}^{(k+1)} + a_3(\phi_1)_{i,j-1}^{(k+1)} + a_2[(\phi_1)_{i+1,j}^{(k)} + (\phi_1)_{i,j+1}^{(k)}] \\ &\qquad\qquad\qquad\qquad\qquad\qquad = (\bar{f}_1)_{i,j}, \\ &-(b_1+2b_2+b_3)(\phi_2)_{i,j}^{(k+1)} + b_1(\phi_2)_{i-1,j}^{(k+1)} + b_3(\phi_2)_{i,j-1}^{(k+1)} + b_2[(\phi_2)_{i+1,j}^{(k)} + (\phi_2)_{i,j+1}^{(k)}] \\ &\qquad\qquad\qquad\qquad\qquad\qquad = (\bar{f}_2)_{i,j}. \end{aligned}\right. \tag{5.12}$$

Define the error functions by $e_1^{(k)} = \phi_1 - \phi_1^{(k)}$ and $e_2^{(k)} = \phi_2 - \phi_2^{(k)}$. Then using (5.11) and (5.12) with freezed $(\bar{f}_1)_{i,j}$ and $(\bar{f}_2)_{i,j}$, the error equations are

$$\left\{\begin{aligned} &a_1(e_1)_{i-1,j}^{(k+1)} + a_3(e_1)_{i,j-1}^{(k+1)} + a_2[(e_1)_{i+1,j}^{(k)} + (e_1)_{i,j+1}^{(k)}] - (a_1+2a_2+a_3)(e_1)_{i,j}^{(k+1)} = 0 \\ &b_1(e_2)_{i-1,j}^{(k+1)} + b_3(e_2)_{i,j-1}^{(k+1)} + b_2[(e_2)_{i+1,j}^{(k)} + (e_2)_{i,j+1}^{(k)}] - (b_1+2b_2+b_3)(e_2)_{i,j}^{(k+1)} = 0. \end{aligned}\right. \tag{5.13}$$

Recall that the LFA measures the largest amplification factor in a relaxation scheme [14, 39, 118]. Let a general Fourier component be

$$\Theta_{\alpha,\beta}(x_i, y_j) = \exp\left(\mathrm{i}\theta_\alpha \frac{x_i}{h} + \mathrm{i}\theta_\beta \frac{y_j}{h}\right) = \exp\left(\frac{2\mathrm{i}\alpha i\pi}{m} + \frac{2\mathrm{i}\beta j\pi}{m}\right).$$

Note that $\theta_\alpha, \theta_\beta \in [-\pi, \pi]$. The LFA expands

$$e_1^{(k)} = \sum_{\alpha,\beta=-m/2}^{m/2} (\psi_1^{(k)})_{\alpha,\beta}\Theta_{\alpha,\beta}(x_i, y_j), \quad e_2^{(k)} = \sum_{\alpha,\beta=-m/2}^{m/2} (\psi_2^{(k)})_{\alpha,\beta}\Theta_{\alpha,\beta}(x_i, y_j)$$

in Fourier components. We look for the largest spectral radius (maximum eigenvalue) of the amplification matrix $\mathcal{A}_{\alpha,\beta}$ [31, 118]:

$$\begin{bmatrix} (\psi_1^{(k+1)})_{\alpha,\beta} \\ (\psi_2^{(k+1)})_{\alpha,\beta} \end{bmatrix} = \mathcal{A}_{\alpha,\beta} \begin{bmatrix} (\psi_1^{(k)})_{\alpha,\beta} \\ (\psi_2^{(k)})_{\alpha,\beta} \end{bmatrix}.$$

After substituting these components into (5.13) for $e_1^{(k+1)}, e_1^{(k)}$ and $e_2^{(k+1)}, e_2^{(k)}$, we have:

$$\mathcal{A}_{\alpha,\beta} = \begin{bmatrix} \dfrac{a_2\left(e^{\frac{2i\alpha\pi}{m}}+e^{\frac{2i\beta\pi}{m}}\right)}{\left(a_1+2a_2+a_3-a_1e^{\frac{-2i\alpha\pi}{m}}-a_3e^{\frac{-2i\beta\pi}{m}}\right)} & 0 \\ 0 & \dfrac{b_2\left(e^{\frac{2i\alpha\pi}{m}}+e^{\frac{2i\beta\pi}{m}}\right)}{\left(b_1+2b_2+b_3-b_1e^{\frac{-2i\alpha\pi}{m}}-b_3e^{\frac{-2i\beta\pi}{m}}\right)} \end{bmatrix}.$$

At the kth iteration, each rate $\bar{\mu}^{(k)}(i,j) = \max_{\alpha,\beta} \rho(\mathcal{A}_{\alpha,\beta})$ in the high frequency range $(\theta_\alpha, \theta_\beta) \in [-\pi, \pi] \setminus [-\frac{\pi}{2}, \frac{\pi}{2}]$, measuring the effectiveness of a smoother [14], is dependent on $a_\ell, b_\ell, \ell = 1, 2, 3$, which in turn depend on the pixel location (i, j). Therefore we should look for the largest smoothing rate for all i, j (i.e. among all such pixels):

$$\hat{\mu} = \max_{a_1,a_2,a_3,b_1,b_2,b_3} \bar{\mu}^{(k)}(i,j).$$

However, due to the high nonlinearity, we found it useful to define the smoothing rate as the maximum of the above accumulated rates out of all s relaxation steps by

$$\hat{\mu}_s = \max_{i,j} \bar{\mu}^{(1)}(i,j)\bar{\mu}^{(2)}(i,j)\cdots\bar{\mu}^{(s)}(i,j).$$

Clearly for linear equations, where a_ℓ, b_ℓ are constants, $\bar{\mu} = \bar{\mu}^{(k)}$ is a constant so $\hat{\mu}_s = \bar{\mu}^{(s)}$. Here, as a_ℓ, b_ℓ are not constants, with this particular definition, we would allow the possibility of $\bar{\mu}^{(k)}(i,j) \approx 1$ for some i, j, k. As long as $\hat{\mu}_s \ll 1$, we would say a smoother is effective.

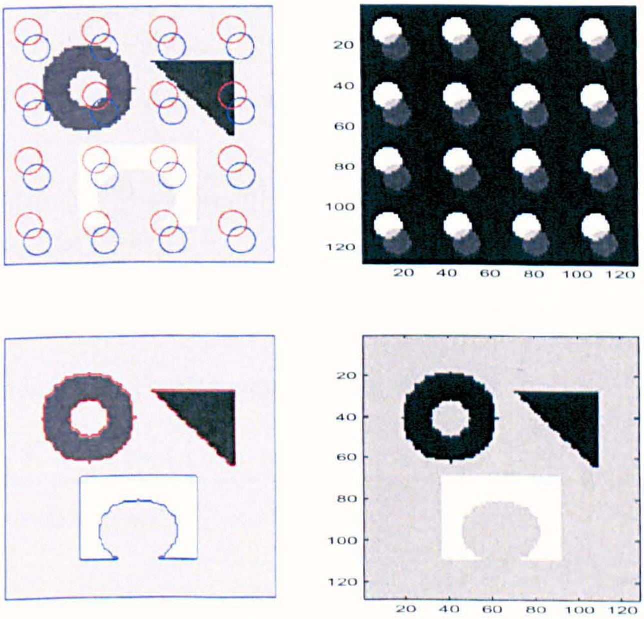


Figure 5.1: Segmentation of the top left image into the bottom right image.

In Table 5.1, we take the particular example of Figure 5.1 of segmenting an artificial image. We use the image size $m = 32$; note that, similar results are obtained with

Table 5.1: The smoothing rate for a local smoother with 3 inner iterations

Outer iterations s	The smoothing rate $\hat{\mu}_s$	The smoothing rate taking out "odd pixels" $\hat{\mu}_s^*$
1	0.6862	0.5720
2	0.6861	0.3170
3	0.6861	0.2747

$m = 64$. Here, in Table 5.1, the "odd pixels" refer to positions where the relative ratios between a_2 and $\max(a_1, a_3)$, or the ratios between b_2 and $\max(b_1, b_3)$, are quite large. Clearly our smoother is ineffective overall due to these odd pixels. This prompted us to consider how to improve the overall smoothing rate (Column 2 in Table 5.1).

A modified smoother. To motivate the idea, consider the particular case of an odd pixel assigned with

$$a_1 = 0.3536, a_2 = 10000, a_3 = 0.3536, b_1 = 0.3536, b_2 = 10000, b_3 = 0.3536 \tag{5.14}$$

for which LFA as described above gives a local (large) rate of $\mu = 0.99996$. Now we propose, as an alternative to (5.12), the following under-relaxation smoothing scheme at these odd pixels:

$$\begin{cases} a_1(\phi_1)_{i-1,j}^{(k+1)} + a_3(\phi_1)_{i,j-1}^{(k+1)} + a_2[(\phi_1)_{i+1,j}^{(k)} + (\phi_1)_{i,j+1}^{(k)}] \\ -(a_1 + 2a_2 + a_3)(1+\omega)(\phi_1)_{i,j}^{(k+1)} + \omega(a_1 + 2a_2 + a_3)(\phi_1)_{i,j}^{(k)} = (\bar{f}_1)_{i,j}, \\ b_1(\phi_2)_{i-1,j}^{(k+1)} + b_3(\phi_2)_{i,j-1}^{(k+1)} + b_2[(\phi_2)_{i+1,j}^{(k)} + (\phi_2)_{i,j+1}^{(k)}] \\ -(b_1 + 2b_2 + b_3)(1+\omega)(\phi_2)_{i,j}^{(k+1)} + \omega(b_1 + 2b_2 + b_3)(\phi_2)_{i,j}^{(k)} = (\bar{f}_2)_{i,j}, \end{cases} \tag{5.15}$$

for some $0 \leqslant \omega \leqslant 1$ (note $\omega = 0$ reduces to the previous local smoother). The new error equation is

$$\begin{cases} a_1(e_1)_{i-1,j}^{(k+1)} + a_3(e_1)_{i,j-1}^{(k+1)} + a_2[(e_1)_{i+1,j}^{(k)} + (e_1)_{i,j+1}^{(k)}] \\ \qquad - (a_1 + 2a_2 + a_3)(1+\omega)(e_1)_{i,j}^{(k+1)} + \omega(a_1 + 2a_2 + a_3)(e_1)_{i,j}^{(k)} = 0, \\ b_1(e_2)_{i-1,j}^{(k+1)} + b_3(e_2)_{i,j-1}^{(k+1)} + b_2[(e_2)_{i+1,j}^{(k)} + (e_2)_{i,j+1}^{(k)}] \\ \qquad - (1+\omega)(b_1 + 2b_2 + b_3)(e_2)_{i,j}^{(k+1)} + \omega(b_1 + 2b_2 + b_3)(e_2)_{i,j}^{(k)} = 0. \end{cases} \tag{5.16}$$

Then the corresponding new Fourier amplification matrix is

$$\mathcal{A}_{\alpha,\beta} = \begin{bmatrix} \dfrac{a_2\left(e^{\frac{2\alpha\pi}{m}} + e^{\frac{2i\beta\pi}{m}}\right) + \omega(a_1 + 2a_2 + a_3)}{\left((1+\omega)(a_1 + 2a_2 + a_3) - a_1 e^{\frac{-2i\alpha\pi}{m}} - a_3 e^{\frac{-2i\beta\pi}{m}}\right)} & 0 \\ 0 & \dfrac{b_2\left(e^{\frac{2i\alpha\pi}{m}} + e^{\frac{2i\beta\pi}{m}}\right) + \omega(b_1 + 2b_2 + b_3)}{\left((1+\omega)(b_1 + 2b_2 + b_3) - b_1 e^{\frac{-2i\alpha\pi}{m}} - b_3 e^{\frac{-2i\beta\pi}{m}}\right)} \end{bmatrix}.$$

Equation (5.15) with $\omega = 0.7$, this new scheme yields a much better rate of $\mu = 0.75026$. The choice of $\omega = 0$ is based on numerical experience.

Therefore, our new smoother will be (5.15) using a variable ω written in a form similar to (5.9) as

$$\begin{aligned}
&D_\ell(\widetilde{\phi_\ell})_{i,j}\Big[(\widetilde{\phi_\ell})_{i+1,j} - (1+\omega)(\phi_\ell)_{i,j} + \omega(\widetilde{\phi_\ell})_{i,j}\Big] \\
&- D_\ell(\widetilde{\phi_\ell})_{i-1,j}\Big[(1+\omega)(\phi_\ell)_{i,j} - \omega(\widetilde{\phi_\ell})_{i,j} - (\widetilde{\phi_\ell})_{i-1,j}\Big] \\
&+ \lambda^2 D_\ell(\widetilde{\phi_\ell})_{i,j}\Big[(\widetilde{\phi_\ell})_{i,j+1} - (1+\omega)(\phi_\ell)_{i,j} + \omega(\widetilde{\phi_\ell})_{i,j}\Big] \\
&\quad - \lambda^2 D_\ell(\widetilde{\phi_\ell})_{i,j-1}\Big[(1+\omega)(\phi_\ell)_{i,j} - \omega(\widetilde{\phi_\ell})_{i,j} - (\widetilde{\phi_\ell})_{i,j-1})\Big] = (\bar{f_\ell})_{i,j}.
\end{aligned} \tag{5.17}$$

It may be stated as follows.

Algorithm 11 (Modified smoother for multiphase model) *Denote a smoothing step for (5.10), using (5.17), by*

$$\phi_\ell^h \longleftarrow Smoother(\phi_\ell^h, \bar{f_\ell}^h, maxit, \omega, K, tol)$$

where $\ell = 1, 2$ *and h is the step size on level* Ω^h. *Set* $K = 100$.

```
for $i = 1 : m_1$
    for $j = 1 : m_2$
        for $iter = 1 : maxit$
        if $|D_\ell(\widetilde{\phi_\ell})^h_{i,j}| \geqslant K \max(|D_\ell(\widetilde{\phi_\ell})^h_{i-1,j}|, |D_\ell(\widetilde{\phi_\ell})^h_{i,j-1}|)$ for any $\ell$, set $\omega = 0.7$;
            otherwise set $\omega = 0$.
            $\widetilde{\phi_\ell}^h \leftarrow \phi_\ell^h$,
            $A_\ell = D_\ell(\widetilde{\phi_\ell})^h_{i,j}((\widetilde{\phi_\ell})^h_{i+1,j} + \omega(\widetilde{\phi_\ell})^h_{i,j}) + D_\ell(\widetilde{\phi_\ell})^h_{i-1,j}((\widetilde{\phi_\ell})^h_{i-1,j} + \omega(\widetilde{\phi_\ell})^h_{i,j})$,
            $B_\ell = D_\ell(\widetilde{\phi_\ell})^h_{i,j}((\widetilde{\phi_\ell})^h_{i,j+1} + \omega(\widetilde{\phi_\ell})^h_{i,j}) + D_\ell(\widetilde{\phi_\ell})^h_{i,j-1}((\widetilde{\phi_\ell})^h_{i,j-1} + \omega(\widetilde{\phi_\ell})^h_{i,j})$,
            $(\phi_\ell)^h_{i,j} = \dfrac{A_\ell + \lambda^2 B_\ell - \bar{f}_{\ell_{i,j}}}{(1+\omega)(D_\ell(\widetilde{\phi_\ell})^h_{i,j} + D_\ell(\widetilde{\phi_\ell})^h_{i-1,j} + \lambda^2(D_\ell(\widetilde{\phi_\ell})^h_{i,j} + D_\ell(\widetilde{\phi_\ell})^h_{i,j-1}))}$
        if $|(\phi_\ell)^h_{i,j} - (\widetilde{\phi_\ell})^h_{i,j}| < tol$ Stop
        end
    end
end
```

We now repeat the smoothing analysis as was done in Table 5.1 and show the new results in Table 5.2. Clearly the new rates are much more acceptable (note the accumulated number of smoothing steps is $3s$ since we use 3 inner iterations for each outer step). In Section 5.5, we shall compare the performance of the two smoothers in MG1.

Table 5.2: The smoothing rate for a modified local smoother

Outer	The smoothing rate
iterations s	$\hat{\mu}_s$
1	0.5720
2	0.3170
3	0.2747

5.4 Multigrid algorithm II for multiphase segmentation (MG2)

As previously remarked, a time marching solution scheme is employed in the original work of Vese-Chan [120]. Realizing that this scheme is extremely slow to converge, Jeon et al [72] proposed a hierarchical image segmentation method which essentially abandoned this multiphase model in favour of the earlier Chan-Vese [35] model. The idea of Jeon et al [72] is the following. We first use the two-phase model [35] to segment the given image z into two phases (a domain and its complement) using a single level set function ϕ. We then segment one of the phases using the two-phase model [35] again and this process is repeated until the desired number of phases is archived. Here there are two key decisions made: (i) The domain having the larger intensity variation will be the next segmentation target; (ii) The domain having the smaller intensity variation will be replaced by a uniform intensity equal to the average intensity of the larger intensity domain. This gives rise to a new image z_{new}, a modified image of z. This new image z_{new} will be segmented. Here the purpose of (ii) is to avoid re-segmenting the domain with the smaller intensity variation.

The aim of this section is to combine the multigrid algorithm [8] for two-phase segmentation with this unsupervised hierarchical image segmentation algorithm [72] and then to assess if any advantage can be gained over Algorithm 10. Our motivation stems from an observation in [8] that the multigrid algorithm can help reach a global minimizer of a two-phase model (i.e. less dependent on initial guesses) while it is not true with the multiphase model (using Algorithm 10).

We first review the important definition used in steps (i-ii) above and then present the combined hierarchical multigrid algorithm. We shall denote by

$$[\phi, c_1, c_2] \leftarrow MGM(\phi, z)$$

the process of utilizing the multigrid algorithm [8] to segment a given image z by working out the desired level set function ϕ, and the two associated constants c_1, c_2.

Definition 5.4.1 (Intensity Variation [72]) *Let z be the given image and S_1, S_2 denote a partition of z, obtained by segmentation using one level set function. Then the intensity variation across S_i is given by*

$$Var(S_i) = \frac{1}{M_i}\sum_{\ell=1}^{M_i}(z(x_\ell, y_\ell) - C_\ell)^2,$$

where C_ℓ represents the average intensity of S_i and M_i is the number of pixels in S_i.

Algorithm 12 (Hierarchical segmentation by multigrid method)

Let n be the required number of segmentation phases and z the given image.

Assume ϕ_0 is an initial contour (which can be a simple pattern or may be worked out by a full multigrid idea as in [8]).

```
for $i = 1, \dots, n-1$
        $\phi_i \leftarrow \phi_0$
        $[\phi_i, c_{i1}, c_{i2}] \leftarrow MGM(\phi_i, z)$, using the 2-phase multigrid method.
            Define $S_1 = \{(k,\ell) \mid (\phi_i)_{k,\ell} < 0\}$ and $S_2 = \{(k,\ell) \mid (\phi_i)_{k,\ell} \geqslant 0\}$.
              Compute $Var(S_1), Var(S_2)$.
            Find $j = \text{argmin}_\ell Var(S_\ell)$ and denote $q = \{1,\ 2\}\backslash\{j\}$.
        Save $C_i = c_{ij}$ and $w_i = j$        (Note $j, q = 1$ or 2.)
      Set $z(S_j) = c_{iq}$ since $S_q$ is the domain with the larger variation.
```

```
end
Set $\phi_n = \phi_{n-1}$, $C_n = c_{iq}$, $w_n = q$.
```

We remark that the main algorithm presented in [72, p.1465] has a major typo, where 'smallest' should mean 'largest', and also the loop should end at $n-1$ rather than n.

Finally once the algorithm is completed, the segmented image will be

$$u = \sum_{j=1}^{n} C_j \Big[(w_j - 1) - (1 - H(\phi_i)) \Big] (-1)^{w_j}, \tag{5.18}$$

which is similar to the two-phase case with $n = 2$, and $w_j = 1, 2$ from Algorithm 12. For instance, if $w_1 = 1, w_2 = 2, w_3 = 1, w_4 = 2$ for SEG4 (i.e. $n = 4$), then

$$u = (1 - H(\phi_1))C_1 + H(\phi_2)C_2 + (1 - H(\phi_3))C_3 + H(\phi_4)C_4.$$

Clearly for n phases, we now require altogether $n - 1$ level set functions, while the previous multiphase method [120] only requires $\log_2 n$ level set functions. For small n, the difference is small; however for large n MG2 will have to store many level set functions (matrices) than MG1.

5.5 Numerical Results

In this section we present experimental results to illustrate the two multigrid algorithms (MG1 and MG2) versus the time marching method i.e.

MG1 – Algorithm 10 with the local smoother (Algorithm 9);
MG1m – Algorithm 10 with the modified local smoother (Algorithm 11);
MG2 – Algorithm 12 with the hierarchical segmentation
AOS – The additive operator splitting method (time-marching).

We shall first compare the qualitative results of segmentation and then compare these solvers in speed of segmentation (iteration steps and CPU time). Although we have done many test examples, we show one artificial image and one real life image here, as shown in Figure 5.2.

Segmentation results. The main parameter μ in the segmentation model balances the regularization term and the fitting term (fitting a phase domain with its average gray-scale levels). Here for Problems 1 and 2 we take μ = (size of image)$^2/12$ and μ = (size of image)$^2/500$ respectively. To detect small objects in an image, smaller the value of μ required and only for larger objects large value of μ is required.

In Figure 5.3, Problem 1 (an image size of 128 x 128) is solved with MG1, MG1m, MG2 and AOS. Even for this small image, we give these computational details to get an impression of these methods: MG1 takes 15 iterations (MG cycles) with CPU time of 12 seconds. MG1m takes 10 iterations with CPU time of 9 seconds. MG2 takes 10 iterations with CPU time of 10 seconds. In the last row final results with AOS are obtained in 230 iterations with CPU time of 64 seconds. Clearly all segmented images are similar to each other while all MG algorithm performances are similar to each other and are faster than AOS.

In Figure 5.4, Problem 2 is solved with MG1, MG1m, MG2 and AOS. The same summary can be made.

In Figure 5.6, we show that MG1 and MG1m does not work very well, and can get wrong solution with some initial guesses. In other words we say that the model (5.2 can stuck at local minima because of its non-convexity. Left figure is the result with MG1

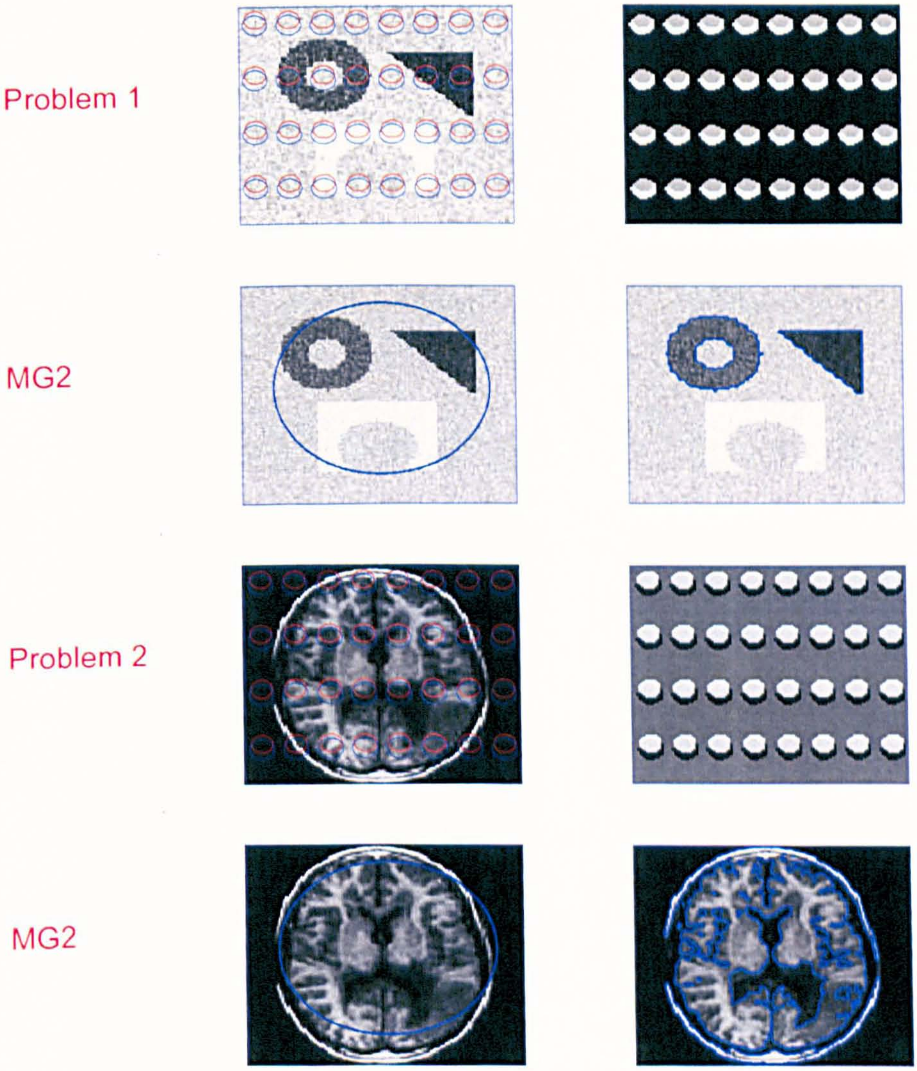


Figure 5.2: Test Problems 1 and 2 with the initial guess contours for MG1, MG1m, AOS methods. For MG2, the initial guesses are for a two-phase model.

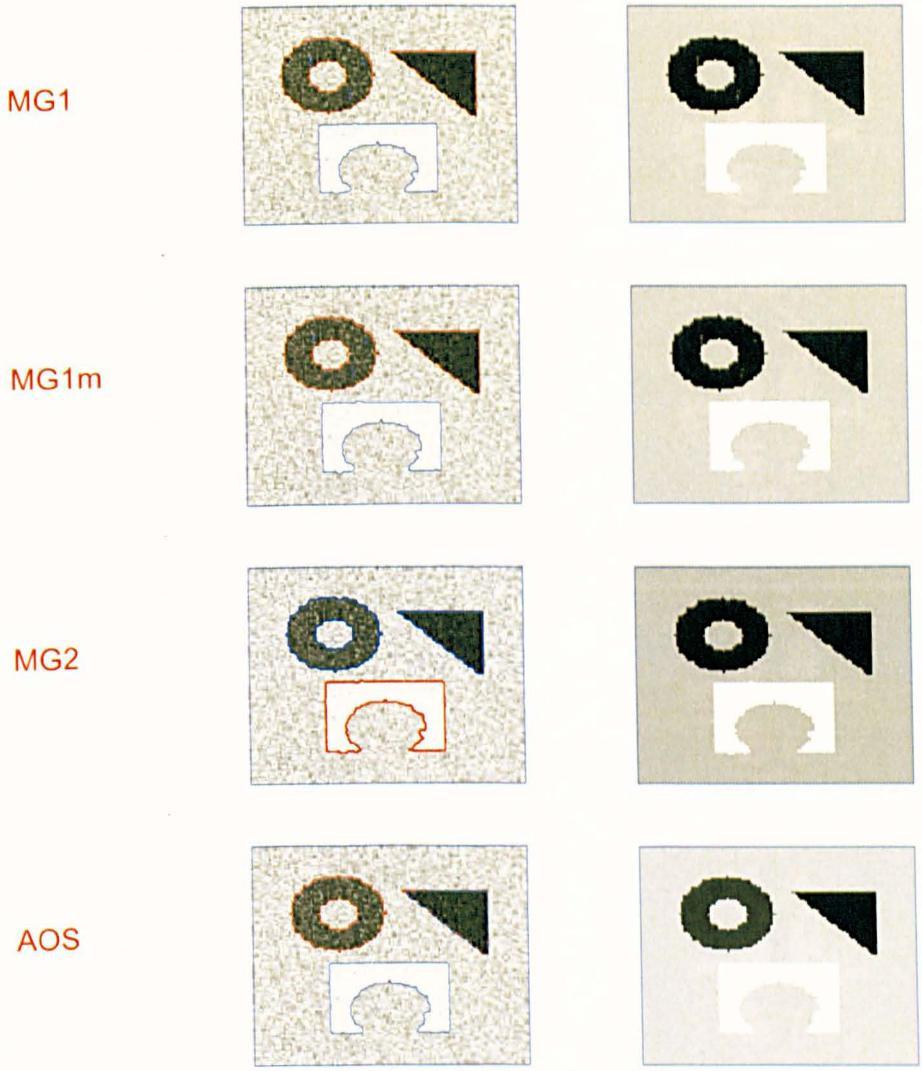


Figure 5.3: Problem 1 solved by MG1, MG1m, MG2, AOS methods. Top row: results with MG1; row 2: results with MG1m; row 3: the results with MG2; and row 4: results with AOS. In row 3 the left image is the three phase segmentation and the right image is the final segmented image.

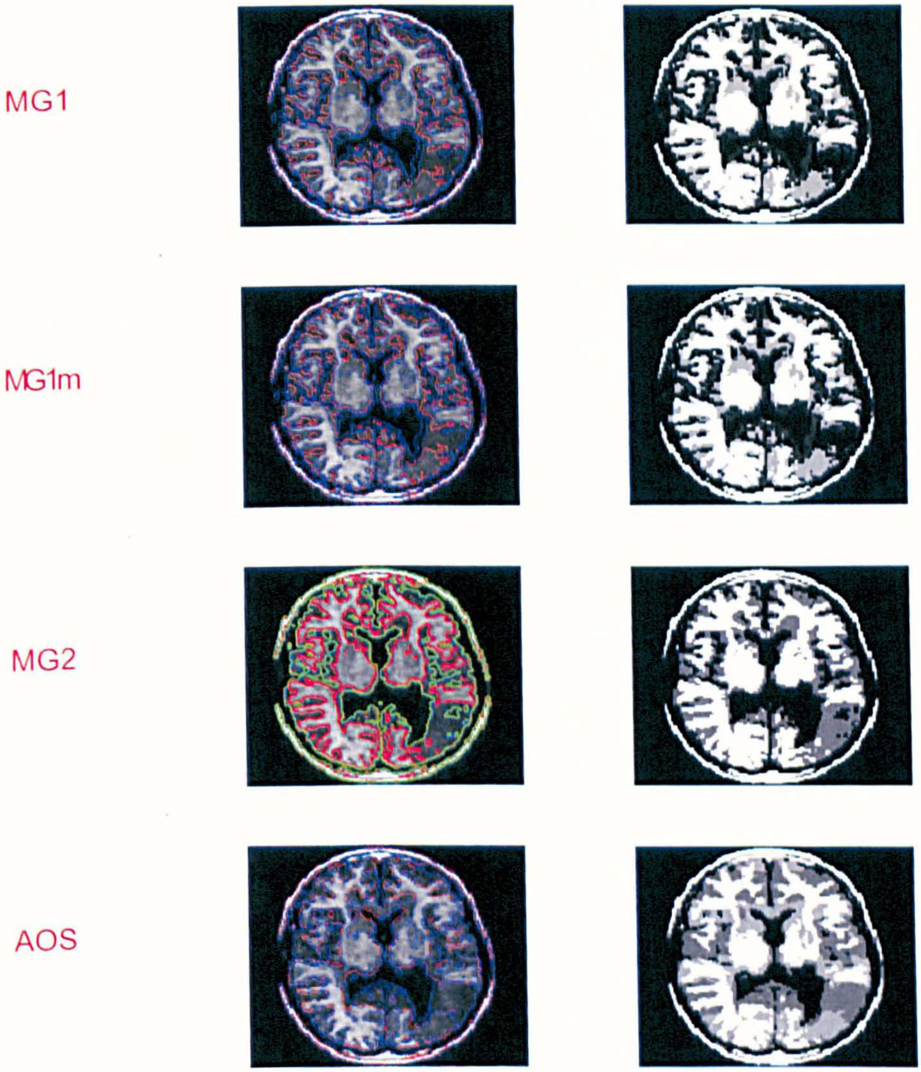


Figure 5.4: Problem 2 solved by MG1, MG1m, MG2, AOS methods. As with Figure 5.3, Top row: results with MG1, row 2: results with MG1m, row 3: the results with MG2 and row 4: results with AOS. In row 3 the left image is the three phase segmentation and the right image is the final segmented image.

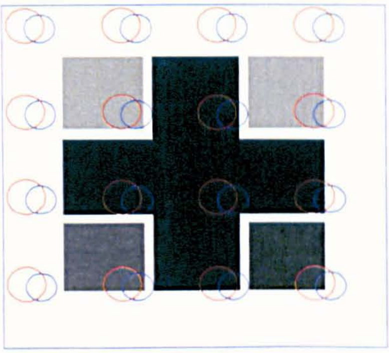


Figure 5.5: Initial guess, where MG1 and MG1m fails to get the desired results.

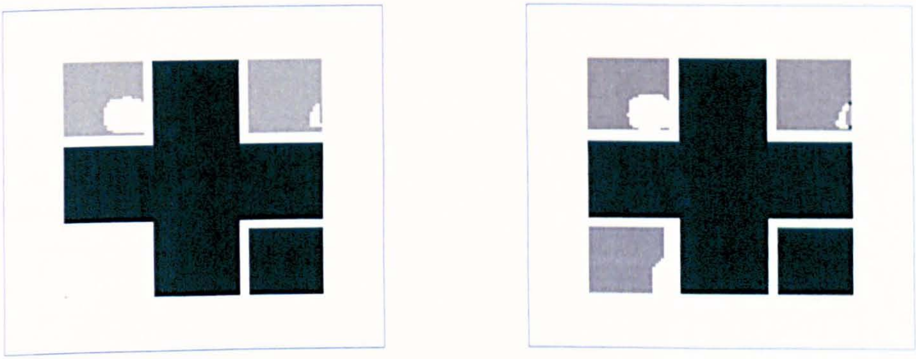


Figure 5.6: Final results from MG1 and MG1m with condition given in Fig 5.5: Left: Results with MG1. Right: Results with MG1m.

and the right figure is the result with MG1m. Clearly we could see that the results from MG1m is better than the results from MG1. In Figure 5.7, gives the results obtained using MG2. Left figure gives the original data and the right figure gives the final results, which is satisfactory. We have tried different initial guesses and got the results.

Performance comparison. In Table 5.3 we compare the methods discussed in this chapter, by CPU times tested on the image in Fig 5.1 in different (larger) sizes. With MG2, in the 8th column of the table we use the notation $p(q)$ implying that p iterations used for the first segmentation and q iterations for the second segmentation. Even for images of small sizes, there is some benefit in using MG algorithms. However, for large sizes, one could see a huge difference in CPU times, with MG algorithms outperforming the AOS by many order of magnitude. MG algorithms yields a computation time of $\mathcal{O}(N \log N)$ where $N = m_1 \times m_2$ [84]. This can be seen in figure 5.8.

5.6 Conclusions

In this chapter we have introduced two multigrid algorithms for multiphase variational image segmentation. As expected of a multigrid method, both algorithms are much

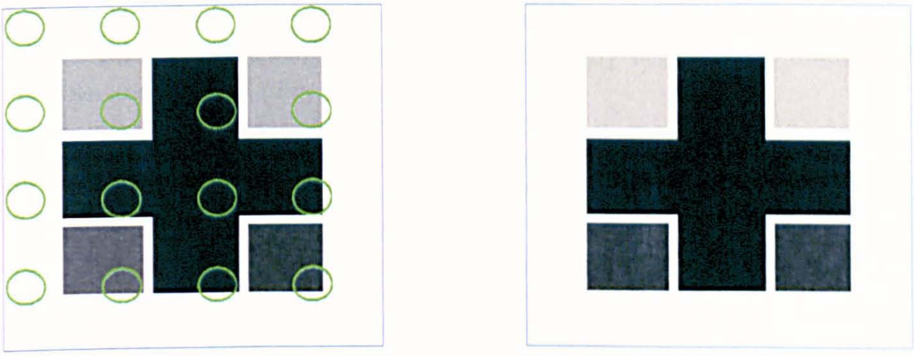


Figure 5.7: Final results using MG2. Left: Initial guess for MG2. Right: Final results with MG2.

Table 5.3: Comparison of MG1, MG1m and MG2 with AOS methods in number of iterations ('Itr') and CPU time ('CPU'). Here '—' implies no convergence (to the tolerance) was achieved with 24 hours.

Image Size	AOS		MG1		MG1m		MG2	
	Itr	CPU	Itr	CPU	Itr	CPU	Itr	CPU
128×128	80	22	3	5	2	2	2(2)	11
256×256	150	193	4	13	2	7	2(2)	15
512×512	1500	42600	4	74	2	33	3(3)	43
1024×1024	—	—	4	525	2	148	3(3)	154

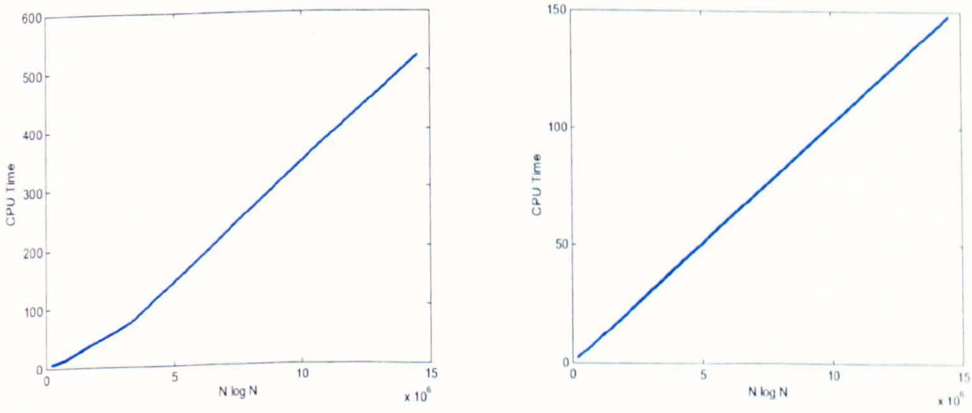


Figure 5.8: MG algorithms yields computation time of $\mathcal{O}(N \log N)$. Left: MG1 CPU vs $N \log N$. Right: MG1m CPU vs $N \log N$

faster than the Additive Operator Splitting (AOS) method. To deliver an acceptable segmentation, MG1 requiring less level set functions can be dependent of initial guesses while MG2 requiring more level set functions is practically independent on the initial guess. Future work will consider models that require only one level set function [4, 78] as well as other models [21, 109].

Chapter 6

An Optimization-based Multilevel Method for Variational Image Segmentation Models

In this chapter we develop an optimization based multilevel method for the CV model discussed in chapter 4 for image segmentation. In this chapter we mainly concern about developing multilevel method for optimization CV models [35], and then will extend it to the models of T. F. Chan et el [25] and X. Bresson et al [15]. We get an improved minimizer with multilevel method and found this is as fast as AOS method compared by CPU time.

6.1 Introduction

The variational formulation has become a well established technique for modeling a class of image processing problems [3, 12, 26, 99]. In chapter 4, we have developed a fast solver (in particular multigrid method) to solve the associated nonlinear partial differential equation (PDE), from the Euler-Lagrange solution of the CV model [35]. Other methods used for solving the PDE are also discussed in chapter 4.

In this chapter we propose an alternative to the PDE approach to solve the image minimization problem directly. The problems to overcome in this study include the treatment of the non-differentiable functional in the minimization problems, by using local minimization (and hence dimension reduction), and the stagnation problem with the primal relaxation, by using coarse levels in a multilevel scheme.

6.2 Model I: The Chan-Vese Two-phase (CV2) Image Segmentation Model:

In level set formulation Chan and Vese [35] proposed the following model (details can be found in Section 3.5)

$$\begin{aligned} F(\phi, c_1, c_2) &= \mu.length(\phi = 0) + \lambda_1 \int_{\phi>0} |z(x,y) - c_1|^2 dxdy \\ &+ \lambda_2 \int_{\phi<0} |z(x,y) - c_2|^2 dxdy. \\ &= \mu \int_\Omega |\nabla H(\phi)|\, dxdy + \lambda_1 \int_\Omega |z(x,y) - c_1| H(\phi) dxdy \\ &+ \lambda_2 \int_\Omega |z(x,y) - c_2|^2 (1 - H(\phi)) dxdy. \end{aligned} \tag{6.1}$$

Keeping ϕ fixed and minimizing equation (6.1) with respect to c_1 and c_2 we have

$$\begin{cases} c_1 = \dfrac{\int_\Omega z(x,y) H(\phi) dxdy}{\int_\Omega H(\phi) dxdy}, & \text{with } \int_\Omega H(\phi) dxdy > 0 \\ c_2 = \dfrac{\int_\Omega z(x,y)(1 - H(\phi)) dxdy}{\int_\Omega (1 - H(\phi)) dxdy}, & \text{with } \int_\Omega (1 - H(\phi)) dxdy > 0. \end{cases} \tag{6.2}$$

In order to formally differentiate (6.1), introduce the regularized Heaviside and Dirac delta functions:

$$H_\epsilon(x) = \frac{1}{2}(1 + \frac{2}{\pi} \arctan(\frac{x}{\epsilon})) \quad \text{and} \quad \delta_\epsilon(x) = H'_\epsilon(x) = \frac{\epsilon}{\pi(x^2 + \epsilon^2)}.$$

The regularized functional F_ϵ of F becomes

$$\begin{aligned} F_\epsilon(\phi, c_1, c_2) &= \mu \int_\Omega |\nabla H_\epsilon(\phi)| dxdy + \lambda_1 \int_\Omega |z(x,y) - c_1|^2 H_\epsilon(\phi) dxdy \\ &+ \lambda_2 \int_\Omega |z(x,y) - c_2|^2 (1 - H_\epsilon(\phi)) dxdy. \end{aligned} \tag{6.3}$$

For given ϕ we compute $c_1(\phi)$ and $c_2(\phi)$ using equation (6.2) and then keep them fixed to update ϕ and then using this new ϕ to update $c_1(\phi)$ and $c_2(\phi)$ and so on.

Variations of (6.3) with respect to ϕ (details can be found in chapter 3) gives the following gradient descent scheme

$$\frac{\partial \phi}{\partial t} = H'_\epsilon(\phi) \Big\{ \mu \nabla . \Big(\frac{\nabla \phi}{|\nabla \phi|} \Big) - (\lambda_1 (z(x,y) - c_1)^2 - \lambda_2 (z(x,y) - c_2)^2) \Big\}. \tag{6.4}$$

In this chapter we develop a new type of multilevel method applied directly to the minimization problem (6.3) to update ϕ. In the next section we first develop optimization based multilevel method for CV (two phase) model (Model I).

6.3 An Optimization Multilevel method (ML1) for CV2

In (6.3), suppose we work out c_1, c_2 first using equation (6.2). Then to update ϕ, consider the following problem

$$\begin{aligned} \min_\phi F_\epsilon(\phi) &= \mu \int_\Omega |\nabla \phi| \delta_\epsilon(\phi)\, dxdy + \lambda_1 \int_\Omega |z(x,y) - c_1|^2 H_\epsilon(\phi) dxdy \\ &+ \lambda_2 \int_\Omega |z(x,y) - c_2|^2 (1 - H_\epsilon(\phi)) dxdy. \end{aligned} \tag{6.5}$$

Let z be the given image on some finest level with $m_1 \times m_2$ pixels so the discrete problem of equation (6.5) is the following

$$\begin{aligned}
&\min_{\{\phi_{i,j}\}'s} F(\phi_{1,1}, \phi_{2,1}, \dots, \phi_{m_1-1,m_2}, \phi_{m_1,m_2}) \\
= \;& \mu \sum_{i=1}^{m_1-1} \sum_{j=1}^{m_2-1} \sqrt{\left(\frac{\phi_{i+1,j} - \phi_{i,j}}{h}\right)^2 + \left(\frac{\phi_{i,j+1} - \phi_{i,j}}{h}\right)^2} . \delta_\epsilon(\phi_{i,j}) h^2 \\
+ \;& \sum_{i=1}^{m_1-1} \sum_{j=1}^{m_2-1} [\lambda_1 (z_{i,j} - c_1)^2 H_\epsilon(\phi_{i,j}) + \lambda_2 (z_{i,j} - c_2)^2 (1 - H_\epsilon(\phi_{i,j}))] . h^2. \\
= \;& \underline{\mu} \sum_{i=1}^{m_1-1} \sum_{j=1}^{m_2-1} \sqrt{\left(\phi_{i+1,j} - \phi_{i,j}\right)^2 + \left(\phi_{i,j+1} - \phi_{i,j}\right)^2} . \delta_\epsilon(\phi_{i,j}) \\
+ \;& \sum_{i=1}^{m_1-1} \sum_{j=1}^{m_2-1} [\underbrace{\lambda_2 (z_{i,j} - c_1)^2 - \lambda_2 (z_{i,j} - c_2)^2}_{r(x,y)}] H_\epsilon(\phi_{i,j}) + \text{terms independent of } \phi
\end{aligned} \tag{6.6}$$

where $\underline{\mu} = \mu/h$ and the minimization is with respect to ϕ so the last term will be dropped in further calculations.

In this section, we consider standard coarsening of of the continuous optimization problem into $L+1$ levels, i.e $k = 1(finest), 2, \dots, L, L+1(coarsest)$. Different from a geometric multigrid method, each coarse level optimization problem will be directly reformulated from finest level as done in [10, 30, 29] for another imaging problem.

First we consider the fine level local minimization which is essentially a coordinate descent method [30, 24].

6.3.1 The finest level local minimization ($k = 1$)

Let $\tilde{\phi}$ be the current iterate. Then our idea is to solve a series of subproblems of the form

$$\min_C F_\epsilon(\tilde{\phi} + C)$$

where C is a local and piecewise constant function. Consider a particular pixel (i, j). Clearly if only $\phi_{i,j}$ is allowed to vary, we simply consider the local subproblem

$$\begin{aligned}
\min_{\phi_{i,j}} F^{local}(\phi_{i,j}) \;=\;& \underline{\mu} \Big[\sqrt{(\phi_{ij} - \tilde{\phi}_{i+1,j})^2 + (\phi_{ij} - \tilde{\phi}_{i,j+1})^2} \delta_\epsilon(\phi_{i,j}) \\
+ \;& \sqrt{(\phi_{ij} - \tilde{\phi}_{i-1,j})^2 + (\tilde{\phi}_{i-1,j} - \tilde{\phi}_{i-1,j+1})^2} \delta_\epsilon(\tilde{\phi}_{i-1,j}) \\
+ \;& \sqrt{(\phi_{ij} - \tilde{\phi}_{i,j-1})^2 + (\tilde{\phi}_{i,j-1} - \tilde{\phi}_{i+1,j-1})^2} \delta_\epsilon(\tilde{\phi}_{i,j-1}) \Big] + r_{ij} H(\tilde{\phi}_{ij}),
\end{aligned}$$

where $r_{i,j} = \lambda_1 (z_{i,j} - c_1)^2 - \lambda_2 (z_{i,j} - c_2)^2$. Starting from $\phi_{i,j}^{old} = \tilde{\phi}_{i,j}$, we can iterate the following (Richardson type) scheme to obtain an approximation for $\phi_{i,j}$:

$$\phi_{i,j}^{new} = RHS/LHS, \tag{6.7}$$

where

$$\begin{aligned}
RHS &= \underline{\mu}\Big[\frac{(\widetilde{\phi}_{i+1,j}+\widetilde{\phi}_{i,j+1})}{L_1}\delta_\epsilon(\phi^{old}_{i,j}) + \frac{\widetilde{\phi}_{i-1,j}.\delta_\epsilon(\widetilde{\phi}_{i-1,j})}{L_2} + \frac{\widetilde{\phi}_{i,j-1}.\delta_\epsilon(\widetilde{\phi}_{i,j-1})}{L_3}\Big] + r_{i,j}\delta_\epsilon(\widetilde{\phi}_{i,j}),\\
LHS &= \underline{\mu}\Big[\frac{2\delta_\epsilon(\phi^{old}_{i,j})}{L_1} + \frac{2\epsilon L_1}{\pi(\epsilon^2+\phi^{old^2}_{i,j})^2} + \frac{\delta_\epsilon(\widetilde{\phi}_{i-1,j})}{L_2} + \frac{\delta_\epsilon(\widetilde{\phi}_{i,j-1})}{L_3}\Big]\\
&\quad \text{and}\\
L_1 &= \sqrt{(\phi^{old}_{ij}-\widetilde{\phi}_{i+1,j})^2 + (\phi^{old}_{ij}-\widetilde{\phi}_{i,j+1})^2 + \beta}\\
L_2 &= \sqrt{(\phi^{old}_{ij}-\widetilde{\phi}_{i-1,j})^2 + (\widetilde{\phi}_{i-1,j}-\widetilde{\phi}_{i-1,j+1})^2 + \beta}\\
L_3 &= \sqrt{(\phi^{old}_{ij}-\widetilde{\phi}_{i,j-1})^2 + (\widetilde{\phi}_{i,j-1}-\widetilde{\phi}_{i+1,j-1})^2 + \beta},
\end{aligned}$$

and $\gamma > 0$ is a regularizing parameter. Equation (6.7)is usually done for few steps only to update $\widetilde{\phi}_{i,j}$.

6.3.2 The General level k local minimization $(1 < k \leqslant L)$

On a general level k, we consider the following minimization subproblem

$$\min_{C} F(\widetilde{\phi}+C), \tag{6.8}$$

where C is a local and piecewise constant function of support $\tau_k \times \tau_k = 2^{k-1} \times 2^{k-1}$ at each block (i,j) of pixels. Formally we may denote the subproblem on level k by

$$\hat{c} = arg \min_{c\in\mathbb{R}^{\tau_k\times\tau_k}} F(\widetilde{\phi}+I_kB_kc), \qquad C_k = I_kB_k\hat{c}, \tag{6.9}$$

where $B_k : \mathbb{R} \to \mathbb{R}^{\tau_k\times\tau_k}$ duplicates a constant to a block of constants, and $I_k : \mathbb{R}^{\tau_k\times\tau_k} \to \mathbb{R}^{n\times n}$ is the interpolation operator so $C_k \in \mathbb{R}^{n\times n}$. Here we may illustrate $C_k = I_kB_k\hat{c}$ as follows [30]

$$C_k = \left[\begin{array}{cc|ccc|cc}
0 & 0 & \cdots & \cdots & \cdots & 0 & 0\\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots\\ \hline
0 & \cdots & c & \cdots & c & \cdots & 0\\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots\\
0 & \cdots & c & \cdots & c & \cdots & 0\\ \hline
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots\\
0 & 0 & \cdots & \cdots & \cdots & 0 & 0
\end{array}\right] \text{ to approximate } \left[\begin{array}{cc|ccc|cc}
c_{11} & \cdots & \cdots & \cdots & \cdots & \cdots & c_{1n}\\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots\\ \hline
c_{i1} & \cdots & c_{ii} & \cdots & c_{ij} & \cdots & c_{in}\\
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots\\
c_{j1} & \cdots & c_{ji} & \cdots & c_{jj} & \cdots & c_{jn}\\ \hline
\cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots\\
c_{n1} & \cdots & \cdots & \cdots & \cdots & \cdots & c_{nn}
\end{array}\right].$$

Next we give some details of solving the local minimization subproblem (6.9). Set on level k, $b = \tau_k = 2^{k-1}, k_1 = (i-1)b+1, k_2 = ib, \ell_1 = (j-1)b+1, \ell_2 = jb$. Firstly we shall note that on level k, there are only $m_1/\tau_k \times m_2/\tau_k$ subproblems each of which is essentially one dimensional (mimicking a coarsegrid of a geometric multigrid method). Secondly we shall show the Richardson type iterative method adopted for each subproblem.

At each block (i,j) of pixels, we solve (6.9) for $c_{i,j}$. Observe that each TV term $|\nabla\phi|$ does not change within the interior pixels of each block on level k because

$$\begin{aligned}
&\sqrt{[(c_{i,j}+\widetilde{\phi}_{k,\ell})-(c_{i,j}+\widetilde{\phi}_{k+1,\ell})]^2 + [(c_{i,j}+\widetilde{\phi}_{k,\ell})-(c_{i,j}+\widetilde{\phi}_{k,\ell+1})]^2}\\
&\qquad = \sqrt{[\widetilde{\phi}_{k,\ell}-\widetilde{\phi}_{k+1,\ell}]^2 + [\widetilde{\phi}_{k,\ell}-\widetilde{\phi}_{k,\ell+1}]^2} \equiv T_{k,\ell}.
\end{aligned}$$

So it remains to consider the contribution to the TV term stemming from the boundary pixels (of the block) and the contribution of all interior pixels to the δ_ϵ term. Thus solving (6.9) is equivalent to solving the following (i,j) block local minimization problem

$$\begin{aligned}
&\min_{c_{i,j}} F(\widetilde{\phi}_{i,j} + I_k B_k c_{i,j}) \\
= \;& \underline{\mu} \sum_{\ell=\ell_1}^{\ell_2} \sqrt{[c_{i,j} - (\widetilde{\phi}_{k_1-1,\ell} - \widetilde{\phi}_{k_1,\ell})]^2 + [\widetilde{\phi}_{k_1-1,\ell} - \widetilde{\phi}_{k_1-1,\ell+1}]^2}.\delta_\epsilon(c_{i,j} + \widetilde{\phi}_{k_1-1,\ell}) \\
+ \;& \underline{\mu} \sum_{k=k_1}^{k_2-1} \sqrt{[c_{i,j} - (\widetilde{\phi}_{k,l_2+1} - \widetilde{\phi}_{k,\ell_2})]^2 + [\widetilde{\phi}_{k,\ell_2} - \widetilde{\phi}_{k+1,\ell_2}]^2}.\delta_\epsilon(c_{i,j} + \widetilde{\phi}_{k,\ell_2}) \\
+ \;& \underline{\mu} \sqrt{[c_{i,j} - (\widetilde{\phi}_{k_2,\ell_2+1} - \widetilde{\phi}_{k_2,\ell_2})]^2 + [c_{i,j} - (\widetilde{\phi}_{k_2+1,\ell_2} - \widetilde{\phi}_{k_2,\ell_2})]^2}.\delta_\epsilon(c_{i,j} + \widetilde{\phi}_{k_2,\ell_2}) \\
+ \;& \underline{\mu} \sum_{\ell=\ell_1}^{\ell_2-1} \sqrt{[c_{i,j} - (\widetilde{\phi}_{k_2+1,\ell} - \widetilde{\phi}_{k_2,\ell})]^2 + [\widetilde{\phi}_{k_2,\ell} - \widetilde{\phi}_{k_2,\ell+1}]^2}.\delta_\epsilon(c_{i,j} + \widetilde{\phi}_{k_2,\ell}) \qquad (6.10) \\
+ \;& \underline{\mu} \sum_{k=k_1}^{k_2} \sqrt{[c_{i,j} - (\widetilde{\phi}_{k,\ell_1-1} - \widetilde{\phi}_{k,\ell_1})]^2 + [\widetilde{\phi}_{k,\ell_1-1} - \widetilde{\phi}_{k+1,\ell_1-1}]^2}.\delta_\epsilon(c_{i,j} + \widetilde{\phi}_{k,\ell_1}) \\
+ \;& \sum_{k=k_1+1}^{k_2-1} \sum_{\ell=\ell_1+1}^{\ell_2-1} T_{k,\ell}.\delta_\epsilon(c_{i,j} + \widetilde{\phi}_{k,\ell}) + \sum_{\ell=\ell_1}^{\ell_2} \sum_{k=k_1}^{k_2} r(k,\ell) H_\epsilon(c_{i,j} + \widetilde{\phi}_{k,\ell}).
\end{aligned}$$

To simplify the formulae, let

$$\Phi_{k,\ell} = \widetilde{\phi}_{k,\ell+1} - \widetilde{\phi}_{k,\ell}, \qquad \Theta_{k,\ell} = \widetilde{\phi}_{k+1,\ell} - \widetilde{\phi}_{k,\ell},$$

and

$$P_{k,\ell} = \frac{\Phi_{k,\ell} + \Theta_{k,\ell}}{2}, \qquad Q_{k,\ell} = \frac{\Phi_{k,\ell} - \Theta_{k,\ell}}{2}.$$

Using the identity

$$\sqrt{(c-a)^2 + (c-b)^2} = \sqrt{2}\sqrt{(c - \frac{a+b}{2})^2 + (\frac{a-b}{2})^2},$$

we may rewrite (6.10) as the following problem

$$\begin{aligned}
\mathcal{F}(c_{i,j}) \;=\;& \underline{\mu} \sum_{\ell=\ell_1}^{\ell_2} \sqrt{(c_{i,j} - \Theta_{k_1-1,\ell})^2 + \Phi^2_{k_1-1,\ell}}.\delta_\epsilon(c_{i,j} + \widetilde{\phi}_{k_1-1,\ell}) \\
+ \;& \underline{\mu} \sum_{k=k_1}^{k_2-1} \sqrt{(c_{i,j} - \Phi_{k,\ell_2})^2 + \Theta^2_{k,\ell_2}}\delta_\epsilon(c_{i,j} + \widetilde{\phi}_{k,\ell_2}) \\
+ \;& \underline{\mu} \sum_{\ell=\ell_1}^{\ell_2-1} \sqrt{(c_{i,j} - \Theta_{k_2,\ell})^2 + \Phi^2_{k_2,\ell}}\delta_\epsilon(c_{i,j} + \widetilde{\phi}_{k_2,\ell}) \\
+ \;& \underline{\mu} \sum_{k=k_1}^{k_2} \sqrt{(c_{i,j} - \Phi_{k,\ell_1})^2 + \Theta^2_{k,\ell_1}}\delta_\epsilon(c_{i,j} + \widetilde{\phi}_{k,\ell_1}) \\
+ \;& \underline{\mu}\sqrt{2}\sqrt{(c_{i,j} - P_{k_2,\ell_2})^2 + (Q_{k_2,\ell_2})^2}\delta_\epsilon(c_{i,j} + \widetilde{\phi}_{k_2,\ell_2}) \\
+ \;& \underline{\mu} \sum_{k=k_1+1}^{k_2-1} \sum_{\ell=\ell_1+1}^{\ell_2-1} T_{k,\ell}.\delta_\epsilon(c_{i,j} + \widetilde{\phi}_{k,\ell}) + \sum_{k=k_1}^{k_2} \sum_{\ell=\ell_1}^{\ell_2} r_{k,\ell} H_\epsilon(c_{i,j} + \widetilde{\phi}_{k,\ell}).
\end{aligned}$$

The first order condition for $\mathcal{F}'(c_{i,j}) = 0$ will take the form

$$\begin{aligned}
\Rightarrow\quad & \underline{\mu}\sum_{\ell=\ell_1}^{\ell_2} \sqrt{(c_{i,j} - \Theta_{k_1-1,\ell})^2 + \Phi^2_{k_1-1,\ell}}.\delta'_\epsilon(c_{i,j} + \widetilde{\phi}_{k_1-1,\ell}) \\
+\quad & \underline{\mu}\sum_{\ell=\ell_1}^{\ell_2} \frac{(c_{i,j} - \Theta_{k_1-1,\ell})}{\sqrt{(c_{i,j} - \Theta_{k_1-1,\ell})^2 + \Phi^2_{k_1-1,\ell}}}.\delta_\epsilon(c_{i,j} + \widetilde{\phi}_{k_1-1,\ell}) + \dots \\
+\quad & \underline{\mu} \sum_{k=k_1+1}^{k_2-1} \sum_{\ell=\ell_1+1}^{\ell_2-1} T_{k,\ell}.\delta'_\epsilon(c_{i,j} + \widetilde{\phi}_{k,\ell}) + \sum_{k=k_1}^{k_2} \sum_{\ell=\ell_1}^{\ell_2} r_{k,\ell}\delta_\epsilon(c_{i,j} + \widetilde{\phi}_{k,\ell}) = 0.
\end{aligned}$$

Since

$$\delta_\epsilon(c_{i,j} + \widetilde{\phi}_{k,\ell}) = \frac{\epsilon}{\pi(\epsilon^2 + (c_{i,j} + \widetilde{\phi}_{k,\ell})^2)}$$

and

$$\delta'_\epsilon(c_{i,j} + \widetilde{\phi}_{k,\ell}) = \frac{-2\epsilon(c_{i,j} + \widetilde{\phi}_{k,\ell})}{\pi(\epsilon^2 + (c_{i,j} + \widetilde{\phi}_{k,\ell})^2)^2},$$

thus we have

$$\begin{aligned}
-\quad & 2\underline{\mu}\sum_{\ell=\ell_1}^{\ell_2} \sqrt{(c_{i,j} - \Theta_{k_1-1,\ell})^2 + \Phi^2_{k_1-1,\ell}}.\frac{(c_{i,j} + \widetilde{\phi}_{k_1-1,\ell})}{(\epsilon^2 + (c_{i,j} + \widetilde{\phi}_{k_1-1,\ell})^2)^2} \\
+\quad & \underline{\mu}\sum_{\ell=\ell_1}^{\ell_2} \frac{(c_{i,j} - \Theta_{k_1-1,\ell})}{\sqrt{(c_{i,j} - \Theta_{k_1-1,\ell})^2 + \Phi^2_{k_1-1,\ell}}}.\frac{1}{(\epsilon^2 + (c_{i,j} + \widetilde{\phi}_{k_1-1,\ell})^2)} + \dots \\
-\quad & 2\underline{\mu} \sum_{k=k_1+1}^{k_2-1} \sum_{\ell=\ell_1+1}^{\ell_2-1} T_{k,\ell}.\frac{(c_{i,j} + \widetilde{\phi}_{k,\ell})}{(\epsilon^2 + (c_{i,j} + \widetilde{\phi}_{k,\ell})^2)^2} + \sum_{k=k_1}^{k_2} \sum_{\ell=\ell_1}^{\ell_2} r_{k,\ell}.\frac{1}{(\epsilon^2 + (c_{i,j} + \widetilde{\phi}_{k,\ell})^2)} = 0.
\end{aligned}$$

Now linearizing the last term in the last equation and solving the following equation for $c^{new}_{i,j} = c_{i,j}$, we get

$$\begin{aligned}
-\quad & 2\underline{\mu}\sum_{\ell=\ell_1}^{\ell_2} \sqrt{(c^{old}_{i,j} - \Theta_{k_1-1,\ell})^2 + \Phi^2_{k_1-1,\ell}}.\frac{(c^{new}_{i,j} + \widetilde{\phi}_{k,\ell})}{(\epsilon^2 + (c^{old}_{i,j} + \widetilde{\phi}_{k_1-1,\ell})^2)^2} \\
+\quad & \underline{\mu}\sum_{\ell=\ell_1}^{\ell_2} \frac{(c^{new}_{i,j} - \Theta_{k_1-1,\ell})}{\sqrt{(c^{old}_{i,j} - \Theta_{k_1-1,\ell})^2 + \Phi^2_{k_1-1,\ell}}}.\frac{1}{(\epsilon^2 + (c^{old}_{i,j} + \widetilde{\phi}_{k_1-1,\ell})^2)} + \dots \\
-\quad & 2\underline{\mu} \sum_{k=k_1+1}^{k_2-1} \sum_{\ell=\ell_1+1}^{\ell_2-1} T_{k,\ell}.\frac{(c^{new}_{i,j} + \widetilde{\phi}_{k,\ell})}{(\epsilon^2 + (c^{old}_{i,j} + \widetilde{\phi}_{k,\ell})^2)^2} \\
+\quad & \sum_{k=k_1}^{k_2} \sum_{\ell=\ell_1}^{\ell_2} r_{k,\ell}.\Big[\frac{-2c^{new}_{i,j}\widetilde{\phi}_{k,\ell}}{(\epsilon^2 + \phi^2_{k,\ell})^2} + \frac{2c^{old}_{i,j}\widetilde{\phi}_{k,\ell}}{(\epsilon^2 + \phi^2_{k,\ell})^2} + \frac{1}{(\epsilon^2 + (c^{old}_{i,j} + \widetilde{\phi}_{k,\ell})^2)}\Big] = 0.
\end{aligned}$$

After some manipulations we get the following iterative scheme for $c^{new}_{i,j}$,

$$c^{new}_{i,j} = RHS^{old}/LHS^{old}, \qquad (6.11)$$

where we start from $c_{i,j}^{old} = 0$,

$$\begin{aligned} RHS^{old} &= 2\underline{\mu}\sum_{\ell=\ell_1}^{\ell_2} \frac{\widetilde{\phi}_{k_1-1,\ell}\sqrt{(c_{i,j}^{old} - \Theta_{k_1-1,\ell})^2 + \Phi_{k_1-1,\ell}^2}}{(\epsilon^2 + (c_{i,j}^{old} + \widetilde{\phi}_{k_1-1,\ell})^2)^2} \\ &+ \underline{\mu}\sum_{\ell=\ell_1}^{\ell_2} \frac{\Theta_{k_1-1,\ell}}{(\epsilon^2 + (c_{i,j}^{old} + \widetilde{\phi}_{k_1-1,\ell})^2)^2\sqrt{(c_{i,j}^{old} - \Theta_{k_1-1,\ell})^2 + \Phi_{k_1-1,\ell}^2}} \\ &+ \ldots + 2\underline{\mu}\sum_{k=k_1+1}^{k_2-1}\sum_{\ell=\ell_1+1}^{\ell_2-1} T_{k,\ell}\cdot\frac{\widetilde{\phi}_{k,\ell}}{(\epsilon^2 + (c_{i,j}^{old} + \widetilde{\phi}_{k,\ell})^2)^2} \\ &- \sum_{k=k_1}^{k_2}\sum_{\ell=\ell_1}^{\ell_2} r_{k,\ell}\cdot\Big[\frac{2c_{i,j}^{old}\widetilde{\phi}_{k,\ell}}{(\epsilon^2 + \phi_{k,\ell}^2)^2} + \frac{1}{(\epsilon^2 + (c_{i,j}^{old} + \widetilde{\phi}_{k,\ell})^2)}\Big], \end{aligned}$$

and

$$\begin{aligned} LHS^{old} &= -2\underline{\mu}\sum_{\ell=\ell_1}^{\ell_2} \frac{\sqrt{(c_{i,j}^{old} - \Theta_{k_1-1,\ell})^2 + \Phi_{k_1-1,\ell}^2}}{(\epsilon^2 + (c_{i,j}^{old} + \widetilde{\phi}_{k_1-1,\ell})^2)^2} \\ &+ \underline{\mu}\sum_{\ell=\ell_1}^{\ell_2} \frac{1}{(\epsilon^2 + (c_{i,j}^{old} + \widetilde{\phi}_{k_1-1,\ell})^2)^2\sqrt{(c_{i,j}^{old} - \Theta_{k_1-1,\ell})^2 + \Phi_{k_1-1,\ell}^2}} \\ &+ \cdots - 2\underline{\mu}\sum_{k=k_1+1}^{k_2-1}\sum_{\ell=\ell_1+1}^{\ell_2-1} \frac{T_{k,\ell}}{(\epsilon^2 + (c_{i,j}^{old} + \widetilde{\phi}_{k,\ell})^2)^2} - 2\sum_{k=k_1}^{k_2}\sum_{\ell=\ell_1}^{\ell_2} r_{k,\ell}\frac{\widetilde{\phi}_{k,\ell}}{(\epsilon^2 + \widetilde{\phi}_{k,\ell}^2)^2}. \end{aligned}$$

Once we have obtained $c_{i,j}$ after a few iterations, $\widetilde{\phi}_{k,\ell}$ is updated by

$$\phi_{k,l} = \widetilde{\phi}_{k,\ell} + c_{i,j}$$

to the full (i,j) block.

6.3.3 The coarsest level minimization ($k = L + 1$)

On the coarsest level the whole image is considered to be a single block, so contribution for updating the constant will only come form the delta function term $\delta_\epsilon(\phi)$, i.e no contribution from the TV term. Thus we consider the following local minimization problem on the coarsest level

$$\min_c F(\widetilde{\phi} + I_k B_k c) = \min_c \underline{\mu}\sum_{i=1}^{m_1}\sum_{j=1}^{m_2} T_{i,j}\delta_\epsilon(\widetilde{\phi}_{i,j} + c) + \sum_{i=1}^{m_1}\sum_{j=1}^{m_2} r_{i,j}H_\epsilon(\widetilde{\phi}_{i,j} + c).$$

Taking variation with respect to c and equating to 0 we have

$$\begin{aligned} &- \underline{\mu}\sum_{i=1}^{m_1}\sum_{j=1}^{m_2} T_{i,j}\frac{\widetilde{\phi}_{i,j} + c^{new}}{(\epsilon^2 + (\widetilde{\phi}_{i,j} + c)^2)^2} \\ &+ \sum_{i=1}^{m_1}\sum_{j=1}^{m_2} r_{i,j}\Big[\frac{2c^{old}\widetilde{\phi}_{i,j}}{(\epsilon^2 + \widetilde{\phi}_{i,j}^2)^2} + \frac{1}{(\epsilon^2 + (c^{old} + \widetilde{\phi}_{i,j})^2)} - \frac{2c^{new}\widetilde{\phi}_{i,j}}{(\epsilon^2 + \widetilde{\phi}_{i,j}^2)^2}\Big] = 0. \qquad (6.12) \end{aligned}$$

Linearizing and solving this equation for c^{new} and then updating $\widetilde{\phi}$ will be similarly done as above.

In summary, we reach at the following algorithm

Algorithm 13 (2D multilevel algorithm (ML1)) $[\phi, c_1, c_2] \leftarrow Opt\,Multilevel1(\phi, z)$
Given the image z and an initial guess $\phi = \phi$ with $L+1$ levels, our multilevel algorithm proceeds as follows:

```
Start
set $\phi_0 = \widetilde{\phi}$ and compute $c_1, c_2$.
for level $k = 1, 2, \ldots, L+1$.
    If $k = 1$, for finest level                    solve(6.7).
    Elseif $k = L+1$ i.e on coarsest level.         solve (6.12) to find c
    Else on all other levels                        solve (6.11).
    Update $\phi = \widetilde{\phi} + I_k B_k c$.
end
Go to Start with $\widetilde{\phi} = \phi$ unless $\|\phi - \phi_0\| < tol$.
```

In fig. 6.1 the initial data are given: original tested images and the initial guesses are shown. We tested with different initial guesses and we get the same results with different initial guess. In fig. 6.2, the results obtained by applying ML1 method to different test images. Left images are the final segmented images and the right images are the original image with final contours. In the top image ML1 is tested on an noisy image which has segmented it very efficiently, where the semi implicit and AOS methods fails to segment it, i.e stuck at local minima [25, 15].

6.4 Model II: Global formulation of CV Model

In this section we discuss the modified model for CV2 and then develop the multilevel method for this new formulation. The CV2 minimization model has a non-convex constraint $H(\phi)$, so the functional (6.1) may get stuck at local minima if one uses the standard algorithms [25] (e.g. a gradient decent method)[1]. This non-convex constraint can be dropped in the following way: The equation (6.4) and the below equation have the same stationary solutions:

$$\frac{\partial \phi}{\partial t} = \mu \nabla . \Big(\frac{\nabla \phi}{|\nabla \phi|}\Big) - (\lambda_1 (z(x,y) - c_1)^2 - \lambda_2 (z(x,y) - c_2)^2). \tag{6.13}$$

Note that c_1 and c_2 are computed for given ϕ using equation (6.2) and then keep them fixed to update ϕ and then update them using updated ϕ. Equation (6.13) is the gradient descent equation for the following energy [25]

$$\mu \int_\Omega |\nabla \phi| + \int_\Omega (\lambda_1 (z(x,y) - c_1)^2 - \lambda_2 (z(x,y) - c_2)^2) \phi dxdy. \tag{6.14}$$

This functional is homogeneous in ϕ of degree 1, as a result it does not have a minimizer in general. If the evolution is carried out for a long time, the level set function ϕ would tends to ∞ in positive direction and $-\infty$ in negative direction. Thus we re-scale ϕ such

[1] We remark that the multigrid method as [8] can find the global minimizer for CV2 and discussed in chapter 4 and so can the ML1 algorithm from the section before.

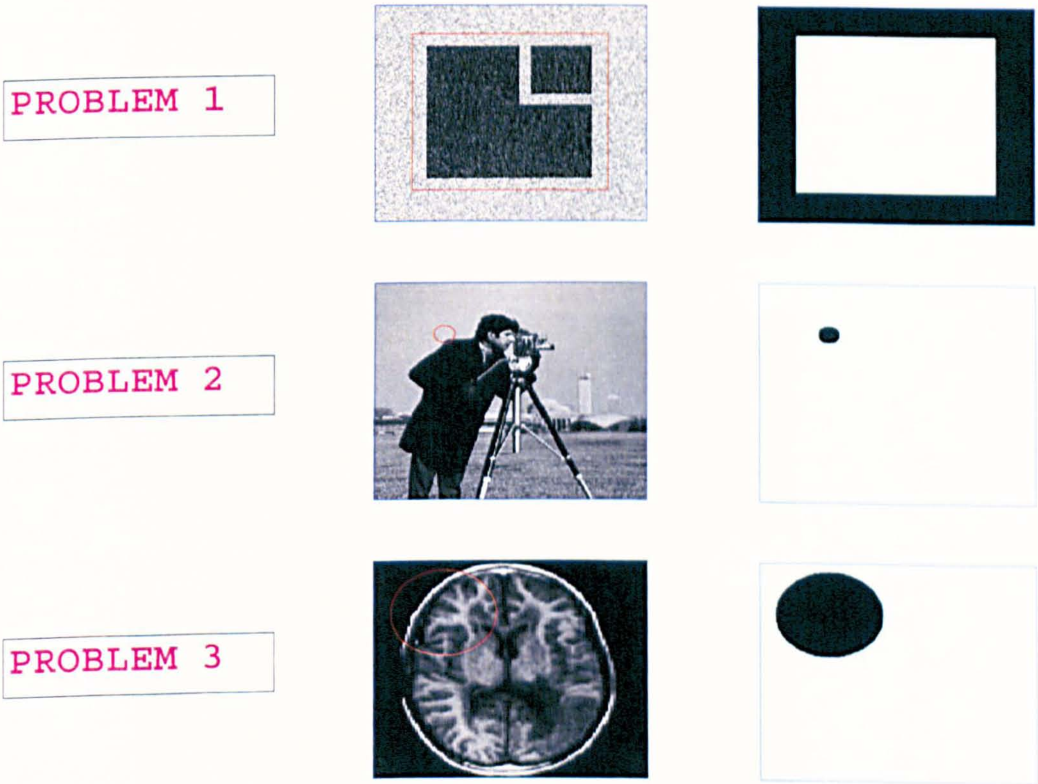


Figure 6.1: Initial data: Left:- Original image for three problems with different initial contour. Right:- Segmented image with different initial contour. Problem 1 is artificial noisy image, problem 2 is a real cameraman image and problem 3 is an real life MRI image

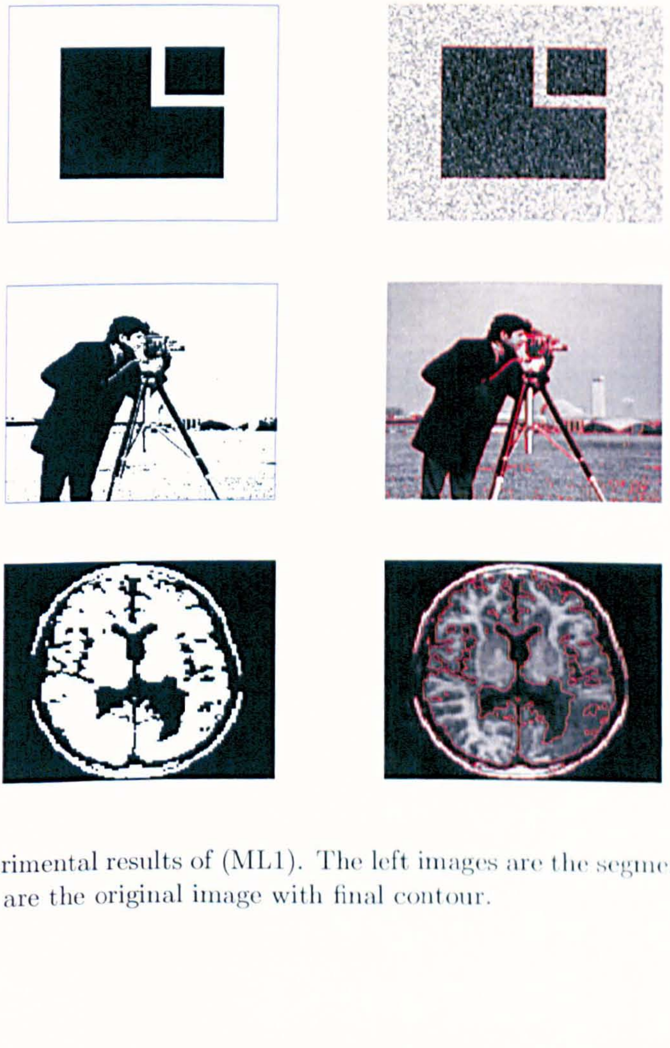


Figure 6.2: Experimental results of (ML1). The left images are the segmented image and the right images are the original image with final contour.

that $-1 \leqslant \phi(\mathrm{x}) \leqslant 1$ for all $\mathrm{x} \in \Omega$ see [25]. Let $r(x,y) = \lambda_1(z(x,y)-c_1)^2 - \lambda_2(z(x,y)-c_2)^2$, then we have the following constraint minimization problem

$$\min_{-1\leqslant\phi\leqslant1} F(\phi) = \mu\int_\Omega |\nabla\phi| + \int_\Omega r(x,y)\phi. \tag{6.15}$$

Euler Lagrange equation for (6.15) is given below

$$div\Big(\frac{\nabla\phi}{|\nabla\phi|}\Big) - r(x,y) = 0, \tag{6.16}$$

where $r(x,y)$ is as above. To solve (6.16),an additive operator splitting (AOS) method [127] has been used previously to solve the following until steady state

$$\frac{\partial\phi}{\partial t} = div\Big(\frac{\nabla\phi}{|\nabla\phi|}\Big) - r(x,y). \tag{6.17}$$

In next section we will solve the minimization problem (6.15) without using variations for ϕ using multilevel method see [30] to get the improved global minima, but c_1 and c_2 will be computed from original CV problem (6.1).

6.4.1 An Optimization Multilevel Method (ML2) for Model II

The discretized form of (6.15), is

$$\begin{aligned}\min_{-1\leqslant\phi_{i,j}\leqslant1} F(\phi(i,j)) \;&=\; \mu\sum_{i=1}^{m_1-1}\sum_{j=1}^{m_2-1}\sqrt{\Big(\frac{\phi_{i+1,j}-\phi_{i,j}}{h}\Big)^2 + \Big(\frac{\phi_{i,j+1}-\phi_{i,j}}{h}\Big)^2}.h^2\\ &+\sum_{i=1}^{m_1-1}\sum_{j=1}^{m_2-1} r_{ij}\phi_{i,j}h^2.\end{aligned} \tag{6.18}$$

Assume that $\tilde{\phi} \in \mathbb{R}^{m_1 \times m_2}$ is the initial approximation to equation (6.16). We look for the best piecewise constant function $C \in \mathbb{R}^{n\times n}$ which minimize the following functional

$$\min_C F(\tilde{\phi} + C). \tag{6.19}$$

First consider C to be local constant at (i,j) and 0 elsewhere (finest level). With the same type of calculations as in previous section, we have the following iterative scheme on fine level

$$\phi_{i,j}^{new} = RHS^{old}/LHS^{old} \tag{6.20}$$

where

$$\begin{aligned} LHS^{old} &= \underline{\mu}\Big(2/L_1 + 1/L_2 + 1/L_3\Big)\\ & \quad\; and\\ RHS^{old} &= \underline{\mu}\Big[(\tilde{\phi}_{i+1,j} + \tilde{\phi}_{i,j+1})/L_1 + \tilde{\phi}_{i-1,j}/L_2 + \tilde{\phi}_{i,j-1}/L_3\Big] - r_{ij}\end{aligned}$$

and

$$\begin{aligned} L_1 &= \sqrt{(\phi_{ij}^{old} - \tilde{\phi}_{i+1,j})^2 + (\phi_{ij}^{old} - \tilde{\phi}_{i,j+1})^2 + \beta}\\ L_2 &= \sqrt{(\phi_{ij}^{old} - \tilde{\phi}_{i-1,j})^2 + (\tilde{\phi}_{i-1,j} - \tilde{\phi}_{i-1,j+1})^2 + \beta}\\ L_3 &= \sqrt{(\phi_{ij}^{old} - \tilde{\phi}_{i,j-1})^2 + (\tilde{\phi}_{i,j-1} - \tilde{\phi}_{i+1,j-1})^2 + \beta,}\end{aligned}$$

where $\gamma > 0$ is a small regularizing parameter. As an iterative method, the scheme (6.20) for (6.19) may still converge to some non-stationary minimizer, so we need to use multilevel method to improve on the obtained minimizer towards getting the global minimum of the functional (6.18).

Second consider C to be a local constant at block $\underbrace{(i,j)}$ on general level k. Similar to ML1, we solve the following equation for $c_{i,j}$ to update ϕ iteratively ($\tilde{\phi} \rightarrow \Phi \rightarrow c_{i,j} \rightarrow \tilde{\phi}\ldots$)

$$(LHS)\mathrm{c_{i,j}} - RHS = 0, \tag{6.21}$$

where

$$\begin{aligned} LHS &= \underline{\mu}\Big(\sum_{\ell=\ell_1}^{\ell_2} 1/\sqrt{(c_{i,j} - \Theta_{k_1-1,\ell})^2 + \Phi^2_{k_1-1,\ell}} + \sum_{k=k_1}^{k_2-1} 1/\sqrt{(c_{i,j} - \Phi_{k,\ell_2})^2 + \Theta^2_{k,\ell_2}} \\ &+ \sum_{\ell=\ell_1}^{\ell_2-1} 1/\sqrt{(c_{i,j} - \Theta_{k_2,\ell})^2 + \Phi^2_{k_2,\ell}} + \sum_{k=k_1}^{k_2} 1/\sqrt{(c_{i,j} - \Phi_{k,\ell_1})^2 + \Theta^2_{k,\ell_1}} \\ &+ \sqrt{2}/\sqrt{(c_{i,j} - P_{k_2,\ell_2})^2 + (Q_{k_2,\ell_2})^2}\Big) \end{aligned}$$

and

$$\begin{aligned} RHS &= \underline{\mu}\Big(\sum_{\ell=\ell_1}^{\ell_2} \Theta_{k_1-1,\ell}/\sqrt{(c_{i,j} - \Theta_{k_1-1,\ell})^2 + \Phi^2_{k_1-1,\ell}} + \sum_{k=k_1}^{k_2-1} \Phi_{k,\ell_2}/\sqrt{(c_{i,j} - \Phi_{k,\ell_2})^2 + \Theta^2_{k,\ell_2}} \\ &+ \sum_{\ell=\ell_1}^{\ell_2-1} \Theta_{k_2,\ell}/\sqrt{(c_{i,j} - \Theta_{k_2,\ell})^2 + \Phi^2_{k_2,\ell}} + \sum_{k=k_1}^{k_2} \Phi_{k,\ell_1}/\sqrt{(c_{i,j} - \Phi_{k,\ell_1})^2 + \Theta^2_{k,\ell_1}} \\ &+ \sqrt{2}P_{k_2,\ell_2}/\sqrt{(c_{i,j} - P_{k_2,\ell_2})^2 + (Q_{k_2,\ell_2})^2}\Big) - \sum_{k=k_1}^{k_2}\sum_{\ell=\ell_1}^{\ell_2} r_{k,\ell}. \end{aligned}$$

Solving for $c_{i,j}$,we arrive at the following Richardson type iteration scheme

$$\mathrm{c_{i,j}} = RHS/LHS. \tag{6.22}$$

RHS and LHS are computed using values $c_{i,j}$ from previous iteration.

As with ML1, on the coarsest Level, the TV term is independent of $c_{i,j}$. We solve the following minimization problem on the coarsest level

$$\min_c \mathcal{F}(c) = \underline{\mu}\sum_{k=1}^{m_1}\sum_{\ell=1}^{m_2}\sqrt{(\tilde{\phi}_{k+1,l} - \tilde{\phi}_{k,\ell})^2 + (\tilde{\phi}_{k,\ell+1} - \tilde{\phi}_{k,\ell})^2} + \sum_{k=1}^{m_1}\sum_{\ell=1}^{m_2} r_{k,\ell}.(\tilde{\phi}_{k,\ell} + c), \tag{6.23}$$

with constraint $-1 \leqslant \tilde{\phi}_{k,\ell} + c \leqslant 1$ or $-(1+\tilde{\phi}_{k,\ell}) \leqslant c \leqslant 1 - \tilde{\phi}_{k,\ell}$. This is a linear functional so has minimum value at one of the end points of the interval, i.e at $c = -(1 + \tilde{\phi}_{k,\ell})$ or $c = 1 - \tilde{\phi}_{k,\ell}$.

Algorithm 14 (2D multilevel algorithm ML2) $[\phi] \leftarrow$ *Opt Multilevel*$(\tilde{\phi}, z, c)$

Given an image z and an initial guess $\tilde{\phi}$ with $L+1$ levels. Our 2D multilevel algorithm ML2 proceeds as follows:

Start

```
set $\phi_0 = \widetilde{\phi}$ and compute $c_1, c_2$.
for level $k = 1, 2, \ldots, L+1$.
        If $k = 1$, on finest level                          solve (6.20).
        Elseif $k = L+1$ i.e on the coarsest level,          solve (6.23).
        Else for other levels,                               solve (6.22).
        Update $\widetilde{\phi} = \widetilde{\phi} + I_k B_k c$.
end
Go to Start with $\widetilde{\phi} = \phi$ unless $\|\phi - \phi_0\| < tol$.
```

We remark that there exist other similarly modified models to the above discussed model which may be solved by adapting ML2. Bresson et al [15] considered the following minimization problem for image segmentation by using weighted TV given below

$$\min_{-1 \leqslant \phi \leqslant 1} F_1(\phi) = \mu \int_\Omega g|\nabla\phi| + \int_\Omega r(x,y)dxdy, \tag{6.24}$$

where $r(x,y)$ is same as defined above and g is edge detector function defined in Eq. (3.10). Minimization of (6.24), Euler Lagrange's equation is given by

$$\frac{\partial\phi}{\partial t} = \mu\nabla.\left(g\frac{\nabla\phi}{|\nabla\phi|}\right) - r(x,y), \tag{6.25}$$

where $-1 \leqslant \phi \leqslant 1$ and $r(x,y) = \lambda_1(z-c_2)^2 - \lambda_2(z-c_2)^2$.

6.5 An Optimization Multilevel Method (ML3) for Multiphase image segmentation

As already discussed in chapter 5 and [9], an effective framework for multiphase segmentation is based on the idea of Jeon et al [72]. We first use the two phase model [35] to segment the given image z into two phases (a domain and its complement) using a single level set function ϕ. We then segment of the phases using the two phase model [35] again and this process is repeated until the desirable number of phases is archived. Here the domain having the larger intensity variation will be the next segmentation target. Details can be found in chapter 5.

In this section we combine our method ML1 with this framework.

Algorithm 15 (Hierarchical segmentation by optimization multilevel (ML1) method)

```
    Let $s$ be the required number of segmentation phases and $z$ the given image.
Assume $\phi_0$ is an initial contour.
for $i = 1, \ldots, s-1$
    $\phi_i \leftarrow \phi_0$
      $[\phi_i, c_{i1}, c_{i2}] \leftarrow$ Opt Multilevel($\phi_i, z, c$) using ML1, using the 2-phase Optimization based multilevel method.
            Define $S_1 = \{(k,\ell) \mid (\phi_i)_{k,\ell} < 0\}$ and $S_2 = \{(k,\ell) \mid (\phi_i)_{k,\ell} \geqslant 0\}$.
              Compute $Var(S_1), Var(S_2)$.
            Find $j = \text{argmin}_\ell Var(S_\ell)$ and denote $q = \{1,\ 2\}\backslash\{j\}$.
        Save the index set $W_i = S_j$.        (Note $j, q = 1$ or $2$.)
        If $i > 1$, find the true index set by modifying $W_i = (W_z \backslash W_{i-1}) \cup W_i$ else continue.
        Set $z(S_j) = c_{iq}$ since $S_q$ is the domain with the larger variation.
    end
    Set the final (phase) index set $W_s = (W_z \backslash W_{s-2}) \cup S_q$.
```

Here assume that W_z denotes the index set of all pixels so the quantity $(W_z \backslash W_{i-1})$ singles out the index set being segmented. Also we can only use the sets S_1, S_2 to identify the domain with the larger variation (to proceed) but its complement (phase i) must be found through $(W_z \backslash W_{i-1})$.

Finally once the algorithm is completed, the segmented image will be separated by the index sets $W_1, W_2, \ldots, W_s$ from which we compute the mean gray values C_j's, by

$$C_j = \sum_{(i,k)\in W_j} z_j / M_j$$

where M_j is the cardinality of W_j. Further the (piecewise) segmented image can be written as $u = (u_{i,k})$ with

$$u_{i,k} = C_j \quad \text{if} \quad (i,k) \in W_j \quad \text{for all} \quad (i,k) \in W_z = W_1 \cup \cdots \cup W_n, \tag{6.26}$$

which is similar to the two-phase case with $s = 2$.

6.6 Results and comparison

In this section we give experimental results of the multilevel and additive operator splitting methods on two models discussed in above sections.

- **ML1:** Optimization based multilevel method for CV model (Model I).
- **ML2:** Optimization based multilevel method for modified CV model (Model II).
- **ML3** Optimization based multilevel level method for the multiphase segmentation.
- **AOS1:** AOS method for the CV model (Model I).
- **AOS2:** AOS method for the modified CV model (Model II).

(1) Qualitative Results. Results for **AOS1** are given in chapter 4 figure 4.12, which stuck at local minima i.e this method can fail to isolate the small box with some initial guesses. In Fig. 6.3 we give the results by using **AOS2** with time step $t = 0.6$. This method is tested on 3 different problems, left figure is the original image with final contour and the right image is the final segmented image. In figure 6.4 results are given from implementation of **ML2**. Figure 6.5, are the results obtained from using AOS solving parabolic PDE (6.25) and Figure 6.6 gives experimental results of multilevel method implemented on minimization model (modified CV model with weighted TV) (6.24).

In Figure (6.7) the implementation of **ML3** for multiphase image segmentation on three different problems is presented. First problem is artificial image need to be segmented into three phases and second problem is the noisy image and is tested for three phase segmentation while the third problem is the real MRI image, segmented in 4-phases. Left figures are the original image with final contour and the right figures are the final segmented images.

(2) Quantitative results. Generally speaking, comparing solutions of different optimizations is not trivial (even though the models are for modeling the same problem), because different functionals are involved. Here we check an obtained solution against all candidate energy functionals (for fairness) and have shown in Table 6.1 the final values of the functionals (6.3) and (6.15). The results in Table 6.1 are for problem 1-2 in Fig. 6.1. Clearly, from Table 6.1, we could see that with our multilevel methods ML1 and ML2 give better results than AOS1 and AOS2 methods.

Note 2 *We use binary images z which normally take values in the range* $[0, 255]$. *The functional (6.3) contains the term* $\lambda_1 \int_\Omega |z - c_1|^2 H(\phi) dxdy + \lambda_2 \int_\Omega |z - c_2|^2 (1 - H(\phi)) dxdy$ *and functional (6.15) contains the term* $\lambda_1 \int_\Omega |z - c_1|^2 dxdy - \lambda_2 \int_\Omega |z - c_2|^2 dxdy$, *which can lead to large values as in table 6.1.*

Table 6.1: Minimum values of the functionals (6.3) and (6.15) using the final ϕ.

Methods/ Functionals	F_ϵ (Model I) Box	F (Model II) Box	F_ϵ (Model I) Cameraman	F (Model II) Cameraman
AOS1 (Model I)	1.6204×10^8	-1.1194×10^8	4.0508×10^7	-4.3651×10^6
ML1 (Model I)	0.0339	-7.8715×10^8	3.9146×10^7	-1.1011×10^9
AOS2 (Model II)	0.0366	-7.5593×10^8	4.8192×10^7	-6.0192×10^8
ML2 (Model II)	0.0333	-7.8638×10^8	3.9146×10^7	-1.1011×10^9

(3) Speed comparisons. Finally we show in Table 6.2 speed information from segmentation of a synthetic and a real image in several resolutions. There the "**" notation indicates that an entry takes too long to obtain a result or the memory requirement is too high to get a result. Clearly our multilevel methods ML1 and ML2 are much faster that AOS type methods.

Table 6.2: Speed comparison of multilevel methods ML1 and ML2 with AOS1 and AOS2 methods.

Problem	Image size	ML1		ML2		AOS1		AOS2	
		Cycles	CPU	Cycles	CPU	Iterations	CPU	Iterations	CPU
Synthetic	128^2	4	1.6	3	1.2	60	4.8	22	1.3
	256^2	4	6.6	3	4.8	140	50	27	7.4
	512^2	5	25.9	3	19.7	280	421	30	36.6
	1024^2	6	107.3	3	80.9	1200	7661	31	155
	2048^2	6	421.6	4	323.95	**	**	**	**
Real	128^2	7	1.7	7	1.3	100	10.6	38	1.6
	256^2	8	6.9	8	5.6	280	110.4	50	13.4
	512^2	9	27.9	9	23	800	1230	55	60.1
	1024^2	12	119.1	12	103.5	**	**	56	288.8
	2048^2	14	497	14	487.12	**	**	**	**

6.7 Conclusion

In this chapter we have presented an optimization based multilevel method for two variational image segmentation models. An mentioned in [25, 15], CV model can easily get

AOS METHOD
WITH TV

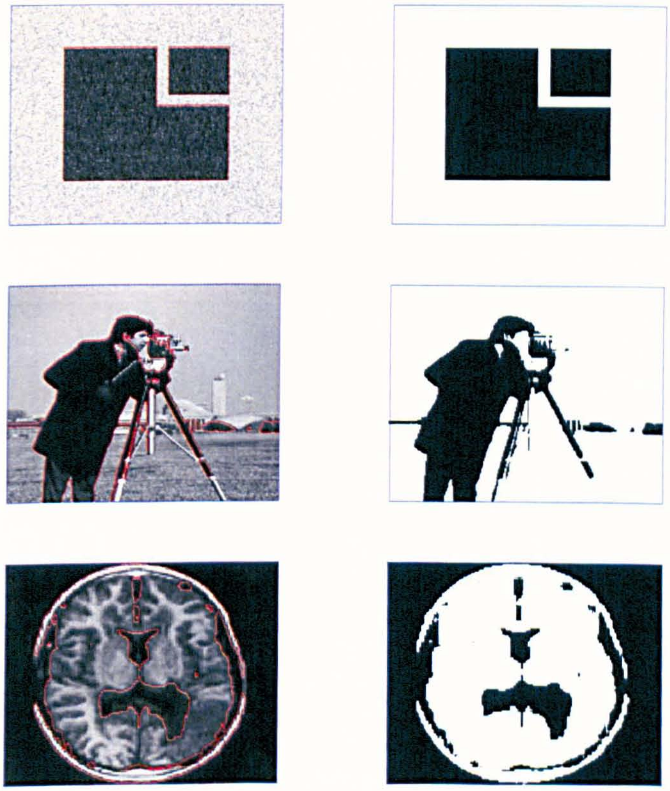


Figure 6.3: Final Results: Left:- Original image with final contour. Right:- Final segmented image. Results from implementation of AOS method for solving parabolic PDE (6.17).

MULTILEVEL
METHOD WITH TV

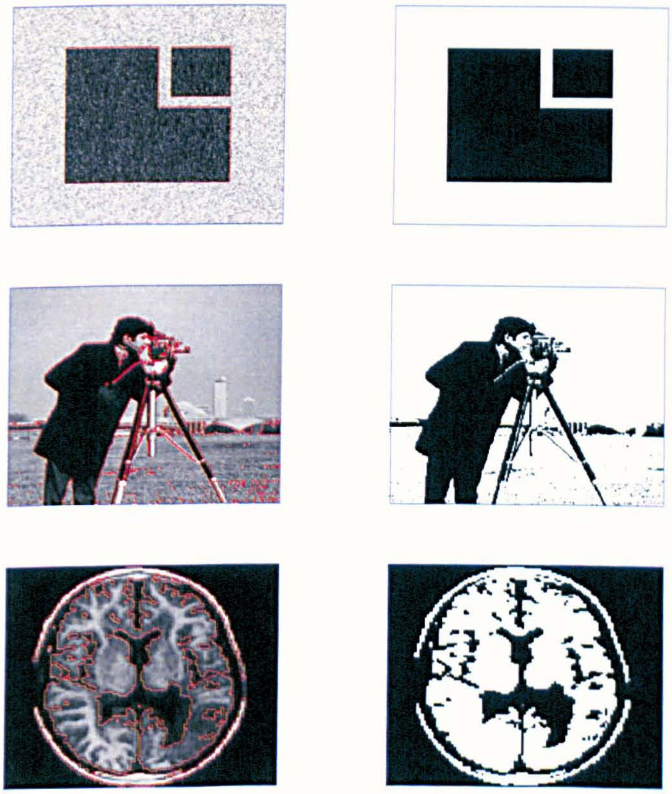


Figure 6.4: Final Results: Left:- Original image with final contour. Right:- Final segmented image. Results from multilevel method for minimization of model (6.15)

AOS METHOD WITH
WEIGHTED TV

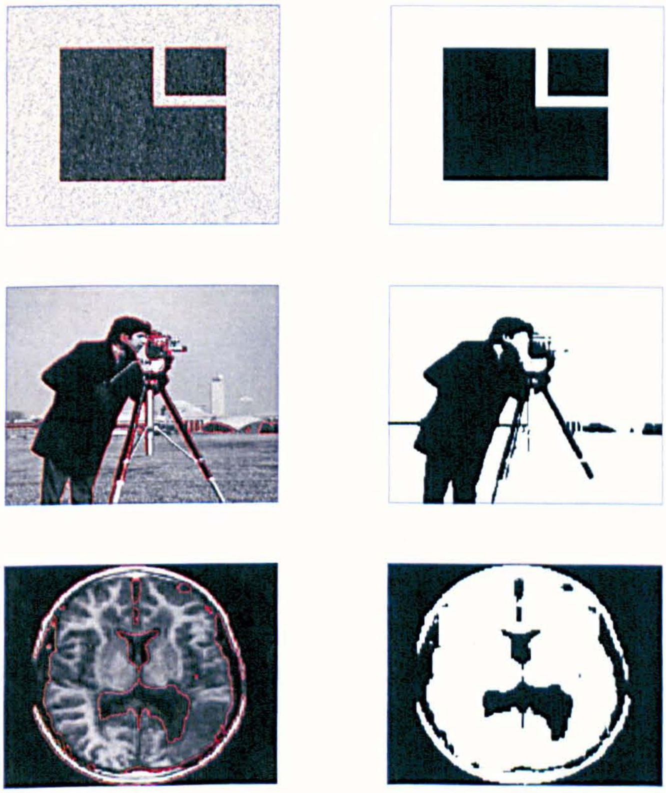


Figure 6.5: Final Results: Left:- Original image with final contour. Right:- Final segmented image. Results obtained from implementation of AOS method to parabolic PDE 6.25.

MULTILEVEL
METHOD WITH
WEIGHTED TV

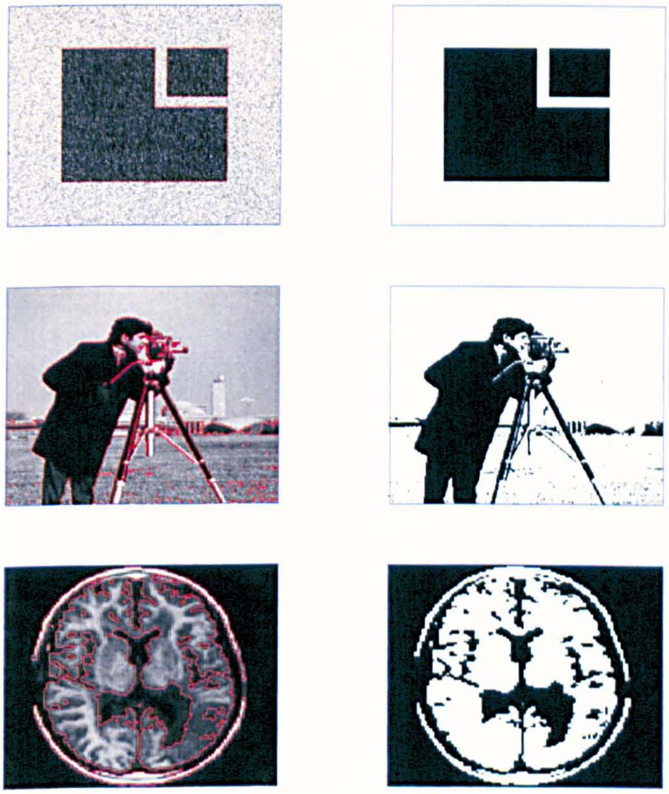


Figure 6.6: Final Results: Left:- Original image with final contour. Right:- Final segmented image. Experimental results from implementing multilevel method to the minimization problem 6.24.

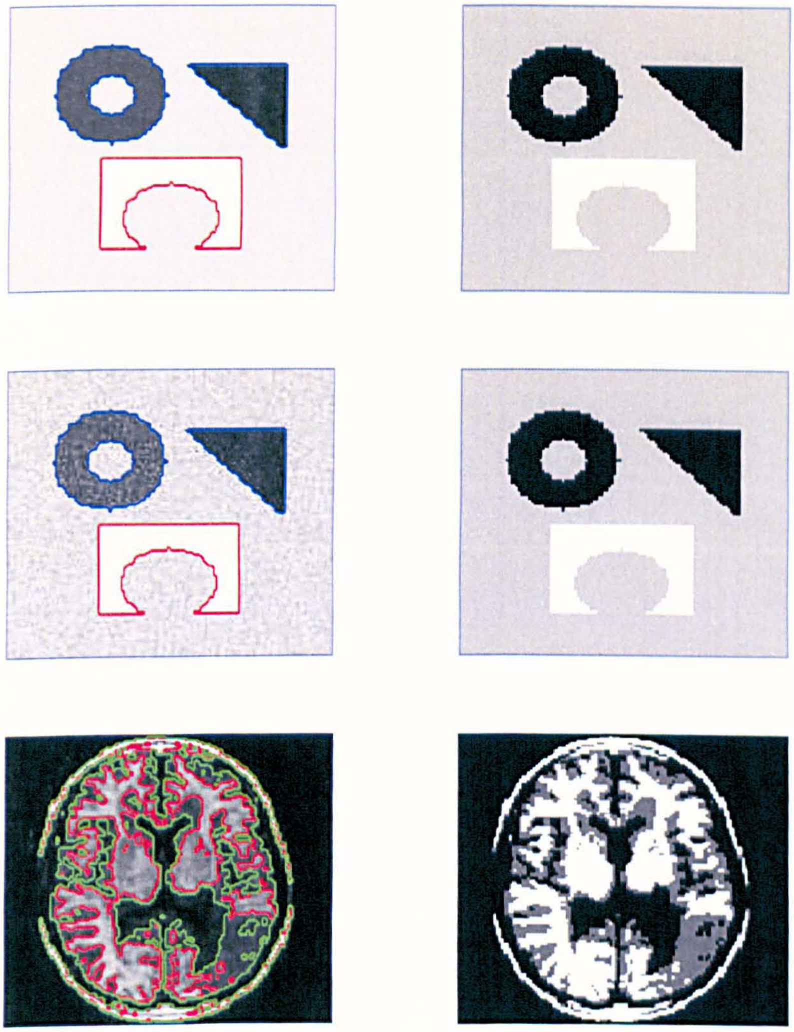


Figure 6.7: Final Results: Left:- Original image with final contour. Right:- Final multiphase segmented image. Experimental results from implementing multilevel method for multiphase image segmentation in hierarchical way.

stuck at local minima discussed in chapter 4, but using this multilevel method ML1 appears to be able to reach the global minimum in all test cases tried, including the problem mentioned in their papers. We applied our developed method to the modified CV model with TV and weighted TV, developed in [25, 15] and found that the new algorithm ML2 can also achieve the global minimum as well as multilevel efficiency. Finally we gave a multiphase segmentation algorithm through repeated use of ML1. Our multilevel algorithms are efficient in speed over other methods like AOS.

Chapter 7

Features Selection in an Image Using an Active Contours Approach

In this chapter we propose a new model for segmentation of an image under some geometrical constraints to automatically detect special features (region of interest ROI) in an image. The model developed by C. Gout et. al in [61, 65] based on the geodesic active contours model can perform such a task in several cases but it is very sensitive to a model parameters. We combine this idea with CV model. The main references for this chapter are [5, 8, 21, 20, 32, 34, 35, 61, 62, 63, 65, 66, 67, 127].

7.1 Introduction

An important problem in image processing is the segmentation of a picture representing a real scene, into classes or categories, corresponding to different objects and the background in the image. In the end, each pixel should belong to one class and only one. In other words, we look for a partition of the image into distinct segments. A variety of different techniques have been developed to solve the problem of image segmentation, such as region growing and emerging [2], watershed algorithms [121], minimum description length criteria [76], and Mumford-Shah energy minimization [88]. Recently, PDE-based active contour models [75] (or curve evolution techniques) have been popular for image segmentation. Some of these methods are discussed in detail in chapter 3. Curve evolution means to evolve deformable contours subject to constraints towards the boundary of the object to be detected. This deformation is made trying to minimize a functional depending on the curve and defined so that a local minimum is obtained at the boundary of the object. Casselles et al [21] have shown, for example, that setting one of the regularization parameters to zero in the classical active contour model is equivalent to finding a geodesic curve in a Riemann space whose metric depends on the image content [65], because an edge in an image is the locus of points for which the image gradient rapidly varies. However, when data acquisition cannot be performed in an optimal manner (e.g the liver in medical imaging), this criterion can no longer be applied. Sometimes the image data is missing or of poor quality, or some occultation occurs or two objects are very close to each other, so have homogeneous intensity, texture etc, and therefore it is hard to clearly identify the interface between them without supplying additional information to a model. Thus the additional information is some geometrical constraints to a model in order to help the segmentation process. Here we consider constraints consisting of a set of points belonging to the contour of interest (the *a priori* knowledge of the physician on the nature and shape of the organ under consideration is therefore crucial, for example liver in medical imaging).

Gout et al [65] proposed a model based on geodesic active contours with in this set of constraints. The Geodesic active contour model uses image gradient information $|\nabla z|$, to stop the contour evolution. This model can only detect objects with edges defined by gradient- otherwise it needs some geometrical constraints. On the other hand, if the given image z is very noisy, then the isotropic smoothing Gaussian has to be strong, which will smooth the edges too. We modify their model by introducing the fitting term of the Chan-Vese model [35], which helps in segmenting noisy images without an isotropic smoothing Gaussian. It also helps in segmenting images with fuzzy boundaries.

This chapter is organized in the following way. Section 7.2 contains a review of the existed model of Gout et al [65]. In Section 7.3 we present our proposed new model and give details of the minimization to get the Euler-Lagrange equation. In Section 7.4 we describe semi implicit method and the AOS method for solving the PDE. In Section 7.5 we give some experimental results.

7.2 Image Segmentation Under Geometrical Conditions (M-1)

We shall first introduce the geometrical conditions before introducing the model by C. Gout et al [65]. Let $z(x,y)$ be the given image defined on a rectangular domain Ω. They have combined the geodesic active contour model [21] with some geometrical constraints such as a set of points near the boundary of object to be detected. Let $A = \{(x_i^*, y_i^*) \in \Omega,\ \ 1 \leqslant i \leqslant n_1\} \subset \Omega$ be the set of n_1 distinct points near the object boundary to be detected in the given image $z(x,y)$. The aim is to find an optimal contour $\Gamma \subset \Omega$ that best approaches the points from the set A while detecting the desire object in an image. To proceed with this, let g be the edge detector function, as defined in equation (3.10), (other forms can be found in [21, 36]). In equation (3.10), the edge detector function is given by

$$g(w) = \frac{1}{1+w^2}.$$

Clearly $g(|\nabla z(x,y)|)$ is zero on edges in an image and is 1 in flat regions. The purpose of the edge detector function g is to stop the evolving curve on edges of the objects in an image. Another function d will be required to stop the evolving curve when approaching the points from set A. Let us define the function d in the following way [65]:

$$\forall (x,y) \in \Omega, \qquad d(x,y) = \prod_{i=1}^{n_1} \left(1 - e^{-\frac{(x-x_i^*)^2}{2\sigma^2}} e^{-\frac{(y-y_i^*)^2}{2\sigma^2}} \right). \qquad (7.1)$$

Other option for d is

$$d(x,y) = distance((x,y), A) = \min_{(x_i^*, y_i^*) \in A} \left| (x,y) - (x_i^*, y_i^*) \right|$$

for all $(x,y) \in \Omega$ and $i = 1, 2, \ldots n_1$ used in [61]. We use the first one. Clearly d acts locally and will be approximately 0 in the neighborhood of points of A. The aim of this model is to find a contour Γ such that $d \simeq 0$ or $g \simeq 0$ along it. They proposed the following energy for this purpose

$$F(\Gamma) = \int_{\Gamma} d(x,y) g(|\nabla z(x,y)|) ds. \qquad (7.2)$$

The contour Γ will stop at local minima where $d \simeq 0$ (in the neighborhood of points for A) or $g \simeq 0$ (near object boundaries).

7.2.1 Level Set Formulation of the Model

To extend the domain of the integral in (7.2) to the whole image other than Γ, they used the level set [92, 90, 108]. Let $\phi : \Omega \rightarrow \mathbb{R}$ be a Lipschitz continuous function. Γ will be considered as the zero level set of ϕ i.e

$$\Gamma = \{(x,y) \in \Omega : \phi(x,y) = 0\},$$

with $\phi < 0$ inside Γ and $\phi > 0$ outside Γ. In terms of the level set formulation, equation (7.2) becomes

$$F(\phi) = \int_{\Omega} d(x,y) g(|\nabla z(x,y)|)|\nabla H(\phi(x,y))| dxdy,$$

where H is the one-dimensional Heaviside function and $\int_{\Omega} |\nabla H(\phi(x,y))| dxdy$ is the length of Γ. Thus we have the following minimization problem

$$\min_{\phi(x,y)} F(\phi(x,y)).$$

Since the Heaviside function is not differentiable at the origin, we consider the regularized version of H denoted by H_ϵ and is defined in equation (3.22). Thus the minimization problem becomes

$$\min_{\phi(x,y)} F_\epsilon(\phi(x,y)), \tag{7.3}$$

where

$$F_\epsilon(\phi(x,y)) = \int_{\Omega} d(x,y) g(|\nabla z(x,y)|) \delta_\epsilon(\phi) |\nabla \phi(x,y)| dxdy. \tag{7.4}$$

Minimization with respect to $\phi(x,y)$ leads to the following Euler-Lagrange equation (details can be found in the next section)

$$-\delta_\epsilon(\phi(x,y)) \nabla \cdot \left(d(x,y) g(|\nabla z(x,y)|) \frac{\nabla \phi(x,y)}{|\nabla \phi(x,y)|} \right) = 0.$$

Gout et al [65] considered the following evolution equation with artificial time step t:

$$\frac{\partial \phi(x,y)}{\partial t} = \delta_\epsilon(\phi(x,y)) \nabla \cdot \left(d(x,y) g(|\nabla z(x,y)|) \frac{\nabla \phi(x,y)}{|\nabla \phi(x,y)|} \right) \tag{7.5}$$

with the boundary condition

$$\frac{\partial \phi(x,y)}{\partial \vec{n}} = 0,$$

where $\vec{n}$ is the outward unit normal to the boundary $\partial\Omega$. Clearly the quantity $\frac{\partial \phi(x,y)}{\partial t}$ tends to 0 when a local minimum is achieved- In other words if the model converges, the curve will not evolve any more since a steady state has been reached. A re-scaling can be made so that the motion is applied to all level sets by replacing $\delta_\epsilon(\phi(x,y))$ by

$|\nabla\phi(x,y)|$. Furthermore, it makes the flow independent of the scaling of ϕ [3, 132]. Thus they considered the following evolution problem

$$\begin{aligned}
\phi(x,y,0) &= \phi_0(x,y), \\
\frac{\partial\phi(x,y)}{\partial t} &= |\nabla\phi(x,y)|\nabla\cdot\left(d(x,y)g(|\nabla z(x,y)|)\frac{\nabla\phi(x,y)}{|\nabla\phi(x,y)|}\right), \\
\frac{\partial\phi(x,y)}{\partial\vec{n}} &= 0 \qquad \text{on } \partial\Omega,
\end{aligned} \tag{7.6}$$

where $\phi_0(x,y)$ is the initial value of $\phi(x,y)$. To speed the convergence of the model they added an extra term $\alpha d(x,y)g(|\nabla z(x,y)|)$ known as a "balloon term" [46] to the evolution equation of a level set, where α is any constant. This term prevents the curve from stopping on a non significant local minimum and is also of importance when initializing the process with a curve inside the object to be detected. Thus the evolution problem becomes

$$\begin{aligned}
\phi(x,y,0) &= \phi_0(x,y) \\
\frac{\partial\phi(x,y)}{\partial t} &= |\nabla\phi(x,y)|\nabla\cdot\left(d(x,y)g(|\nabla z(x,y)|)\frac{\nabla\phi(x,y)}{|\nabla\phi(x,y)|}\right) \\
&+ \alpha d(x,y)g(|\nabla z(x,y)|)|\nabla\phi(x,y)| \\
\frac{\partial\phi(x,y)}{\partial n} &= 0 \qquad \text{on } \partial\Omega.
\end{aligned} \tag{7.7}$$

The main equation in (7.7) can be written as

$$\begin{aligned}
\frac{\partial\phi(x,y)}{\partial t} &= |\nabla\phi(x,y)|d(x,y)g(|\nabla z(x,y)|)\nabla\cdot\left(\frac{\nabla\phi(x,y)}{|\nabla\phi(x,y)|}\right) \\
&+ \nabla(d(x,y)g(|\nabla z(x,y)|))\cdot\nabla\phi + \alpha d(x,y)g(|\nabla z(x,y)|)|\nabla\phi(x,y)|.
\end{aligned} \tag{7.8}$$

The Additive Operator Splitting method [127] was used to solve this evolution problem.

This model is based on geodesic active contours, which use gradient information of the image as discussed above, and curvature to detect the boundary, in which only local information of the boundary is used. Thus it is difficult to get ideal results when dealing with fuzzy edges and discrete edges. Furthermore, because of the local attributes and the dependence on gradient, geodesic active contours are heavily affected by noisy inputs: it is hard to detect objects from a noisy image which is a generic problem with all edge detectors. One can use isotropic Gaussian smoothing $K_\sigma * z$, but this will smooth the edges too, see Figure 7.1, we have used the following filter

$$K_\sigma = 1/9\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \tag{7.9}$$

other options are Gaussian filter or Laplacian. We propose a new model whose stopping term is based on Mumford and Shah segmentation techniques [88]. With this new model we can detect objects in a noisy images without using isotropic Gaussian smoothing.

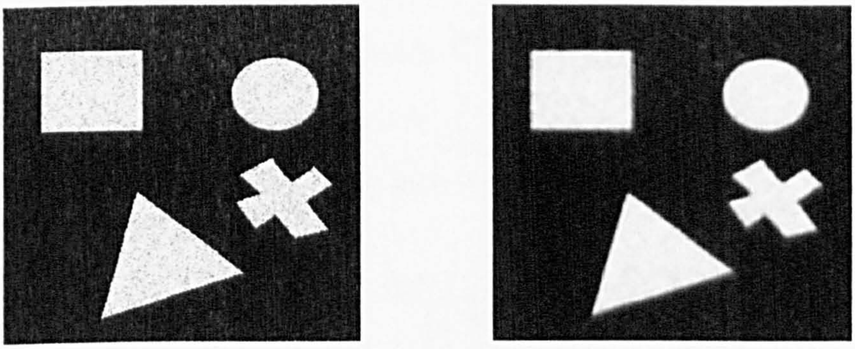


Figure 7.1: Left: Original synthetic noisy image. Right: Smoothing of the original image $K_\sigma * z$.

7.3 Proposed Model (M-2)

The Chan-Vese (CV) model is a special case of the piecewise constant Mumford and Shah model (restricted to only 2 regions). The CV model is not based on the gradient of the image $z(x, y)$ for the stopping process. It can detect contours both with and without gradients. Also there is no need to smooth the image in the case of a noisy image. To use the advantages of the CV model [35], we add $\lambda_1 \int_{inside(\Gamma)} |z(x, y) - c_1|^2 dxdy + \lambda_2 \int_{outside(\Gamma)} |z(x, y) - c_2|^2 dxdy$ to the model (7.2) where λ_1, λ_2 are some constants and c_1, c_2 are average values of the given image $z(x, y)$ inside and outside Γ. Thus we propose the following minimization problem

$$\min_{\phi(x,y),c_1,c_2} F(\phi(x, y), c_1, c_2), \tag{7.10}$$

where

$$\begin{aligned} F(\Gamma, c_1, c_2) = \ & \mu \textstyle\int_\Gamma d(x, y) g(|\nabla z(x, y)|) ds + \\ & \lambda_1 \textstyle\int_{inside(\Gamma)} |z(x, y) - c_1|^2 dxdy + \lambda_2 \int_{outside(\Gamma)} |z(x, y) - c_2|^2 dxdy, \end{aligned} \tag{7.11}$$

where μ is a positive parameter. Clearly if $\lambda_1 = \lambda_2 = 0$ and $\mu = 1$ then minimization problem (7.11) reduces to minimization problem (7.2).

The level set formulation of the functional (7.11) is:

$$\begin{aligned} F(\phi(x, y), c_1, c_2) = \mu & \int_\Omega d(x, y) g(|\nabla z(x, y)|) |\nabla H(\phi(x, y))| dxdy \\ & + \lambda_1 \int_\Omega |z(x, y) - c_1|^2 H(\phi(x, y)) dxdy + \\ & \lambda_2 \int_\Omega |z(x, y) - c_2|^2 (1 - H(\phi(x, y))) dxdy, \end{aligned} \tag{7.12}$$

where H is the Heaviside function.

Again using the regularized Heaviside function H_ϵ, we consider the following minimization problem

$$\min_{\phi(x,y),c_1,c_2} F_\epsilon(\phi(x, y), c_1, c_2), \tag{7.13}$$

where

$$F_\epsilon(\phi(x,y),c_1,c_2) = \mu \int_\Omega d(x,y)g(|\nabla z(x,y)|)\delta_\epsilon(\phi(x,y))|\nabla\phi(x,y)|dxdy +$$

$$\lambda_1 \int_\Omega |z(x,y)-c_1|^2 H_\epsilon(\phi(x,y))dxdy + \lambda_2 \int_\Omega |z(x,y)-c_2|^2(1-H_\epsilon(\phi(x,y)))dxdy. \quad (7.14)$$

Keeping $\phi(x,y)$ fixed and minimizing with respect to c_1 and c_2 we have the following equations for computing c_1 and c_2:

$$c_1(\phi(x,y)) = \frac{\int_\Omega z(x,y)H_\epsilon(\phi(x,y))dxdy}{\int_\Omega H_\epsilon(\phi(x,y))dxdy} \quad (7.15)$$

if $\int_\Omega H_\epsilon(\phi(x,y))dxdy > 0$ (i.e if the curve has a nonempty interior in Ω), and

$$c_2(\phi(x,y)) = \frac{\int_\Omega z(x,y)(1-H_\epsilon(\phi(x,y)))dxdy}{\int_\Omega(1-H_\epsilon(\phi(x,y)))dxdy} \quad (7.16)$$

if $\int_\Omega(1-H_\epsilon(\phi(x,y)))dxdy > 0$ (i.e if the curve has a nonempty exterior in Ω).

Now keeping c_1 and c_2 fixed, we minimize (7.13) with respect to $\phi(x,y)$. To minimize F_ϵ, we use the Gâteaux derivatives to find the first first variation of the functional F_ϵ with respect to ϕ

$$\lim_{h\to 0}\frac{1}{h}\left(F_\epsilon(\phi+h\psi,c_1,c_2) - F_\epsilon(\phi,c_1,c_1)\right) = 0,$$

$$\text{i.e} \qquad \mu\int_\Omega d(x,y)g(|\nabla z(x,y)|)\left(\delta'_\epsilon(\phi)|\nabla\phi|\psi + \delta_\epsilon(\phi)\frac{\nabla\phi\cdot\nabla\psi}{|\nabla\phi|}\right)dxdy$$

$$+ \int_\Omega \delta_\epsilon(\phi)(\lambda_1(z(x,y)-c_1)^2 - \lambda_2(z(x,y)-c_2)^2)\psi dxdy = 0, \quad (7.17)$$

where ψ is a test function of the same type as ϕ. To complete we use ϕ in place of $\phi(x,y)$ in these calculations and similarly for ψ. From Green's Theorem we have

$$\int_\Omega v\nabla.\vec{w}dx = -\int_\Omega \nabla v.\vec{w}dx + \int_{\partial\Omega} v\vec{w}.\vec{n}ds.$$

$$\text{Hence taking } \ \psi = v \qquad \text{and} \qquad G(x,y)\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\nabla\phi = \vec{w},$$

where $G(x,y) = d(x,y)g(|\nabla z(x,y)|)$. We have

$$\int_\Omega \psi\nabla\cdot\Big(G(x,y)\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\nabla\phi\Big)dx = -\int_\Omega \nabla\psi\cdot G(x,y)\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\nabla\phi dx + \int_{\partial\Omega}\psi G(x,y)\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\nabla\phi\cdot\vec{n}ds,$$

implies that

$$\int_\Omega G(x,y)\delta_\epsilon(\phi)\frac{\nabla\phi\cdot\nabla\psi}{|\nabla\phi|}dx = -\int_\Omega \psi\nabla\cdot\left(G(x,y)\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\nabla\phi\right)dx + \int_{\partial\Omega}\psi G(x,y)\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\frac{\partial\phi}{\partial\vec{n}}ds,$$

where $\nabla\phi\cdot\vec{n} = \frac{\partial\phi}{\partial\vec{n}}$. Thus equation (7.17) becomes

$$\int_\Omega \mu G(x,y)\delta'_\epsilon(\phi)|\nabla\phi|\psi dxdy + \int_{\partial\Omega}\mu G(x,y)\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\frac{\partial\phi}{\partial\vec{n}}\psi ds$$

$$- \int_\Omega \mu\nabla\cdot\left(\delta_\epsilon(\phi)G(x,y)\frac{\nabla\phi}{|\nabla\phi|}\right)\psi dxdy$$

$$+ \int_\Omega \delta_\epsilon(\phi)(\lambda_1(z(x,y)-c_1)^2 - \lambda_2(z(x,y)-c_2)^2)\psi dxdy = 0.$$

$$
\begin{aligned}
\Rightarrow \quad & \int_\Omega \mu G(x,y)\delta_\epsilon'(\phi)|\nabla\phi|\psi dxdy + \int_{\partial\Omega} \mu G(x,y)\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\frac{\partial\phi}{\partial\vec{n}}\psi ds \\
- \quad & \int_\Omega \mu\delta_\epsilon(\phi)\nabla\cdot\left(G(x,y)\frac{\nabla\phi}{|\nabla\phi|}\right)\psi dxdy - \int_\Omega \mu\delta_\epsilon'(\phi)G(x,y)\nabla\phi\cdot\frac{\nabla\phi}{|\nabla\phi|}\psi dxdy \\
+ \quad & \int_\Omega \delta_\epsilon(\phi)(\lambda_1(z(x,y)-c_1)^2-\lambda_2(z(x,y)-c_2)^2)\psi dxdy = 0.
\end{aligned}
$$

This gives

$$
\begin{aligned}
& -\int_\Omega \mu\delta_\epsilon(\phi)\nabla\cdot\left(G(x,y)\frac{\nabla\phi}{|\nabla\phi|}\right)\psi dxdy \\
+ \quad & \int_{\partial\Omega} \mu G(x,y)\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\frac{\partial\phi}{\partial\vec{n}}\psi ds \\
+ \quad & \int_\Omega \delta_\epsilon(\phi)(\lambda_1(z(x,y)-c_1)^2-\lambda_2(z(x,y)-c_2)^2)\psi dxdy = 0.
\end{aligned}
$$

This holds for all test functions ψ. Thus we have the following Euler-Lagrange equation for ϕ:

$$
\begin{aligned}
& \delta_\epsilon(\phi)\mu\nabla\cdot\left(G(x,y)\frac{\nabla\phi}{|\nabla\phi|}\right) \\
- \quad & \delta_\epsilon(\phi)(\lambda_1(z(x,y)-c_1)^2-\lambda_2(z(x,y)-c_2)^2) = 0, \qquad \text{on } \Omega \\
& G(x,y)\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\frac{\partial\phi}{\partial\vec{n}} = 0, \qquad \text{on } \partial\Omega.
\end{aligned}
\tag{7.18}
$$

To solve this PDE we consider the following evolution equation

$$
\begin{aligned}
\frac{\partial\phi}{\partial t} &= \delta_\epsilon(\phi)\mu\nabla\cdot\left(G(x,y)\frac{\nabla\phi}{|\nabla\phi|}\right) \\
&- \delta_\epsilon(\phi)(\lambda_1(z(x,y)-c_1)^2-\lambda_2(z(x,y)-c_2)^2)
\end{aligned}
\tag{7.19}
$$

with the boundary condition

$$
G(x,y)\frac{\delta_\epsilon(\phi)}{|\nabla\phi|}\frac{\partial\phi}{\partial\vec{n}}\bigg|_{\partial\Omega} = 0,
$$

where $\vec{n}$ is the unit normal vector to the boundary of Ω. At steady state $\frac{\partial\phi}{\partial t} = 0$, which means the local minimum has been reached. To extend the motion to all level sets one can replace the delta function $\delta_\epsilon(\phi)$ by the gradient $|\nabla\phi|$- this will make the flow independent of the scaling of ϕ [132]. But we would keep $\delta_\epsilon(\phi(x,y))$ in the equation. Thus we consider the following evolution problem

$$
\begin{cases}
\phi(x,y,0) = \phi_0(x,y) \\
\dfrac{\partial\phi}{\partial t} = \mu\delta_\epsilon(\phi(x,y))\nabla\cdot\left(G(x,y)\dfrac{\nabla\phi}{|\nabla\phi|}\right) \\
-\delta_\epsilon(\phi)(\lambda_1(z(x,y)-c_1)^2-\lambda_2(z(x,y)-c_2)^2), \\
G(x,y)\dfrac{\delta_\epsilon(\phi)}{|\nabla\phi|}\dfrac{\partial\phi}{\partial n}\bigg|_{\partial\Omega} = 0.
\end{cases}
\tag{7.20}
$$

A term $\alpha G(x,y)|\nabla\phi|$ (known as a balloon term) could be added to speed up the convergence of the evolution equation as done in model **M-1**, where α is a constant [45]. This term prevents the curve from stopping on a non significant local minimum and is also of importance when initializing the process with a curve inside the object to be detected [65]. Thus equation (7.20) with balloon term can be written as

$$\begin{cases} \phi(x,y,0) = \phi_0(x,y) \\ \dfrac{\partial\phi}{\partial t} = \mu\delta_\epsilon(\phi(x,y))\nabla\cdot\left(G(x,y)\dfrac{\nabla\phi}{|\nabla\phi|}\right) \\ -\delta_\epsilon(\phi)(\lambda_1(z(x,y)-c_1)^2 - \lambda_2(z(x,y)-c_2)^2) + \alpha G(x,y)|\nabla\phi|, \\ G(x,y)\dfrac{\delta_\epsilon(\phi)}{|\nabla\phi|}\dfrac{\partial\phi}{\partial n}\Big|_{\partial\Omega} = 0, \end{cases} \tag{7.21}$$

or

$$\begin{cases} \phi(x,y,0) = \phi_0(x,y) \\ \dfrac{\partial\phi}{\partial t} = \mu\delta_\epsilon(\phi(x,y))G(x,y)\nabla\cdot\left(\dfrac{\nabla\phi}{|\nabla\phi|}\right) + \mu\delta_\epsilon(\phi(x,y))\nabla G(x,y)\cdot\left(\dfrac{\nabla\phi}{|\nabla\phi|}\right) \\ -\delta_\epsilon(\phi)(\lambda_1(z-c_1)^2 - \lambda_2(z-c_2)^2) + \alpha G(x,y)|\nabla\phi|, \\ G(x,y)\dfrac{\delta_\epsilon(\phi)}{|\nabla\phi|}\dfrac{\partial\phi}{\partial n}\Big|_{\partial\Omega} = 0. \end{cases} \tag{7.22}$$

Existence and uniqueness of the solution can be proved along similar lines to [65].

7.4 Numerical Methods

In this section we present some numerical methods for solving PDE (7.22).

7.4.1 Semi-Implicit method

For given ϕ compute $c_1(\phi)$ and $c_2(\phi)$ using equations (7.15) and (7.16) respectively and then keep them fixed to update ϕ using equation (7.22). And then use the new ϕ to update c_1 and c_2 and so on. Let us consider the PDE in (7.22), and let $f(x,y) = \delta_\epsilon(\phi)(-\lambda_1(z-c_1)^2 + \lambda_2(z-c_2)^2) + \alpha G(x,y)|\nabla\phi|$, we have

$$\frac{\partial\phi}{\partial t} = \mu\delta_\epsilon(\phi(x,y))\left[G(x,y)\nabla\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right) + \nabla G(x,y)\cdot\left(\frac{\nabla\phi}{|\nabla\phi|}\right)\right] + f(x,y).$$

Using the differences $\Delta_+^x, \Delta_-^x, \dots$ defined in equation (3.29), the discretized form of equation (7.23) is:

$$\begin{aligned} \frac{\phi_{i,j}^{k+1} - \phi_{i,j}^k}{\Delta t} &= \mu\delta_\epsilon(\phi_{i,j}^k)G_{i,j}\left[\frac{1}{h_1^2}\Delta_-^x\left(\frac{\Delta_+^x\phi_{i,j}^{k+1}}{\sqrt{(\Delta_+^x\phi_{i,j}^k/h_1)^2 + (\Delta_+^y\phi_{i,j}^k/h_2)^2}}\right)\right. \\ &+ \left.\frac{1}{h_2^2}\Delta_-^y\left(\frac{\Delta_+^y\phi_{i,j}^{k+1}}{\sqrt{(\Delta_+^x\phi_{i,j}^k/h_1)^2 + (\Delta_+^y\phi_{i,j}^k/h_2)^2}}\right)\right] \\ &+ \mu\frac{\delta_\epsilon(\phi_{i,j}^k)}{|\nabla\phi_{i,j}^k|}\left\{\frac{1}{h_1^2}\Delta_+^x G(x,y)\Delta_+^x\phi_{i,j}^{k+1} + \frac{1}{h_2^2}\Delta_+^y G(x,y)\Delta_+^y\phi_{i,j}^{k+1}\right\} + f_{i,j}. \end{aligned}$$

We usually use $h_1 = h_2 = 1$, so we have

$$\begin{aligned}
\frac{\phi^{k+1}_{i,j} - \phi^k_{i,j}}{\Delta t} &= \mu\delta_\epsilon(\phi_{i,j}G^k_{i,j})\left[\left(\frac{\Delta^x_+\phi^{k+1}_{i,j}}{\sqrt{(\Delta^x_+\phi^k_{i,j})^2 + (\Delta^y_+\phi^k_{i,j})^2}}\right)\right. \\
&- \left(\frac{\Delta^x_+\phi^{k+1}_{i-1,j}}{\sqrt{(\Delta^x_+\phi^k_{i-1,j})^2 + (\Delta^y_+\phi^k_{i-1,j})^2}}\right) \\
&+ \left(\frac{\Delta^y_+\phi^{k+1}_{i,j}}{\sqrt{(\Delta^x_+\phi^k_{i,j})^2 + (\Delta^y_+\phi^k_{i,j})^2}}\right) \\
&- \left.\left(\frac{\Delta^y_+\phi^{k+1}_{i,j-1}}{\sqrt{(\Delta^x_+\phi^k_{i,j-1})^2 + (\Delta^y_+\phi^k_{i,j-1})^2}}\right)\right] \\
&+ \mu\frac{\delta_\epsilon(\phi^k_{i,j})}{|\nabla\phi^k_{i,j}|}\left\{\Delta^x_+G(x,y)\Delta^x_+\phi^{k+1}_{i,j} + \Delta^y_+G(x,y)\Delta^y_+\phi^{k+1}_{i,j}\right\} + f_{i,j}.
\end{aligned}$$

This implies that

$$\begin{aligned}
\frac{\phi^{k+1}_{i,j} - \phi^k_{i,j}}{\Delta t} &= \mu\delta_\epsilon(\phi^k_{i,j})G_{i,j}\left[\left(\frac{\phi^{k+1}_{i+1,j} - \phi^{k+1}_{i,j}}{\sqrt{(\Delta^x_+\phi^k_{i,j})^2 + (\Delta^y_+\phi^k_{i,j})^2}}\right)\right. \\
&- \left(\frac{\phi^{k+1}_{i,j} - \phi^{k+1}_{i-1,j}}{\sqrt{(\Delta^x_+\phi^k_{i-1,j})^2 + (\Delta^y_+\phi^k_{i-1,j})^2}}\right) \\
&+ \left(\frac{\phi^{k+1}_{i,j+1} - \phi^{k+1}_{i,j}}{\sqrt{(\Delta^x_+\phi^k_{i,j})^2 + (\Delta^y_+\phi^k_{i,j})^2}}\right) \\
&- \left.\left(\frac{\phi^{k+1}_{i,j} - \phi^{k+1}_{i,j-1}}{\sqrt{(\Delta^x_+\phi^k_{i,j-1})^2 + (\Delta^y_+\phi^k_{i,j-1})^2}}\right)\right] \\
&+ \mu\frac{\delta_\epsilon(\phi^k_{i,j})}{|\nabla\phi^k_{i,j}|}\left\{\Delta^x_+G(x,y)\Delta^x_+\phi^{k+1}_{i,j} + \Delta^y_+G(x,y)\Delta^y_+\phi^{k+1}_{i,j}\right\} + f_{i,j}.
\end{aligned}$$

Let

$$\begin{aligned}
D_{i-1,j} &= 1/\sqrt{(\Delta^x_+\phi^k_{i-1,j})^2 + (\Delta^y_+\phi^k_{i-1,j})^2}, \\
D_{i,j} &= 1/\sqrt{(\Delta^x_+\phi^k_{i,j})^2 + (\Delta^y_+\phi^k_{i,j})^2}, \\
D_{i,j-1} &= 1/\sqrt{(\Delta^x_+\phi^k_{i,j-1})^2 + (\Delta^y_+\phi^k_{i,j-1})^2}. \qquad (7.23)
\end{aligned}$$

With the above notation the discretized equation becomes:

$$\begin{aligned}
\frac{\phi^{k+1}_{i,j} - \phi^k_{i,j}}{\Delta t} &= \mu\delta_\epsilon(\phi^k_{i,j})G_{i,j}\Big[(\phi^{k+1}_{i+1,j} - \phi^{k+1}_{i,j})D_{i,j} - (\phi^{k+1}_{i,j} - \phi^{k+1}_{i-1,j})D_{i-1,j} \\
&+ (\phi^{k+1}_{i,j+1} - \phi^{k+1}_{i,j})D_{i,j} - (\phi^{k+1}_{i,j} - \phi^{k+1}_{i,j-1})D_{i,j-1}\Big] \qquad (7.24) \\
&+ \mu\frac{\delta_\epsilon(\phi^k_{i,j})}{|\nabla\phi^k_{i,j}|}\left\{\Delta^x_+G(x,y)(\phi^{k+1}_{i+1,j} - \phi^{k+1}_{i,j}) + \Delta^y_+G(x,y)(\phi^{k+1}_{i,j+1} - \phi^{k+1}_{i,j})\right\} + f_{i,j}.
\end{aligned}$$

As the coefficients $D_{i-1,j}$, $D_{i,j}$ and $D_{i,j-1}$ has been freezed at k, equation (7.24) gives a linear system of equations which can be solved by any iterative method. The implicit method is stable for large time steps but the main drawback is the computational cost [126]. Hence we use Additive Operator Splitting (AOS) [127] to solve the PDE (7.21). The details are given in the next section.

7.4.2 The Additive Operator Splitting (AOS) Method

Let us consider equation (7.21) and let $f = -\delta_\epsilon(\phi)(\lambda_1(z-c_1)^2-\lambda_2(z-c_2)^2)+\alpha G(x,y)|\nabla\phi|$ and $F = \dfrac{G}{|\nabla\phi|}$ we have

$$\frac{\partial\phi}{\partial t} = \mu\delta_\epsilon(\phi)\nabla\cdot(F\nabla\phi) + f. \tag{7.25}$$

The additive operator splitting scheme (AOS) [127] splits the m-dimensional spatial operator into a sum of m one-dimensional space discretizations. Therefore we consider the following one dimensional problem to be solved first

$$\frac{\partial\phi}{\partial t} = \mu\delta_\epsilon(\phi)(\partial_x(F\partial_x\phi) + \partial_y(F\partial_y\phi)) + f, \tag{7.26}$$

and the discretization gives with spatial step size $(h_1, h_2 = 1)$

$$\frac{\phi_i^{k+1} - \phi_i^k}{\Delta t} = \mu\delta_\epsilon(\phi)\left((\frac{F_i^k + F_{i+1}^k}{2})(\phi_{i+1}^{k+1} - \phi_i^{k+1}) - (\frac{F_i^k + F_{i-1}^k}{2})(\phi_i^{k+1} - \phi_{i-1}^{k+1})\right) + f_i, \tag{7.27}$$

$$\Rightarrow \phi_i^{k+1} = \phi_i^k + \mu\Delta t(c_1\phi_{i+1}^{k+1} - c_2\phi_i^{k+1} + c_3\phi_{i-1}^{k+1}) + f_i, \tag{7.28}$$

where

$$c_1 = \delta_\epsilon(\phi)\frac{F_i^k + F_{i+1}^k}{2}, \qquad c_2 = \delta_\epsilon(\phi)\frac{F_{i-1}^k + 2F_i^k + F_{i+1}^k}{2}, \text{ and } c_3 = \delta_\epsilon(\phi)\frac{F_i^k + F_{i-1}^k}{2}.$$

We solve the system of equations (7.28) with double time step Δt in the x, y- direction and then average the two solutions. In matrix notation equation (7.28) can be written as: for $l = 1, 2$ we have

$$(I - 2\Delta t A_l(\phi^k))\phi_l^{k+1} = f^k$$
$$\phi^{k+1} = \frac{1}{2}\sum_{l=1}^{2}\phi_l^{k+1},$$

where I is the identity matrix and A_l for $l = 1, 2$ are tridiagonal matrices derived from (7.28).

7.5 Experimental Results

In this section firstly we present some examples where **M-1** does not work very well. Secondly we show that our model **M-2** works on these examples. We further test our model on real images. Lastly we give evidence that our model **M-2** is faster than **M-1** in convergence in terms of number of iterations and CPU time. In figure 7.2, **M-1** is tested on synthetic noisy image to detect the rectangle in the image with 4 markers shown in the figure with red dots. This model clearly fails to detect the rectangle.

Top left figure is the original image with initial data (red dots are the markers) and top

right is the result after 1000 iterations (hardly any improvement).
Bottom left is the result after 6000 iterations and bottom right is the final result after 20000 iterations where the level set function ϕ does not move any more.
Parameters: $\phi_0 = \sqrt{(x-x_0)^2+(y-y_0)^2} - 25$ where $x_0 = \dfrac{\sum x\text{-comp of markers}}{\text{no. of markers}}$ and $y_0 = \dfrac{\sum y\text{-comp of markers}}{\text{no. of markers}}$ and $\sigma = 4, \alpha = -0.00151, \triangle t = 1$.

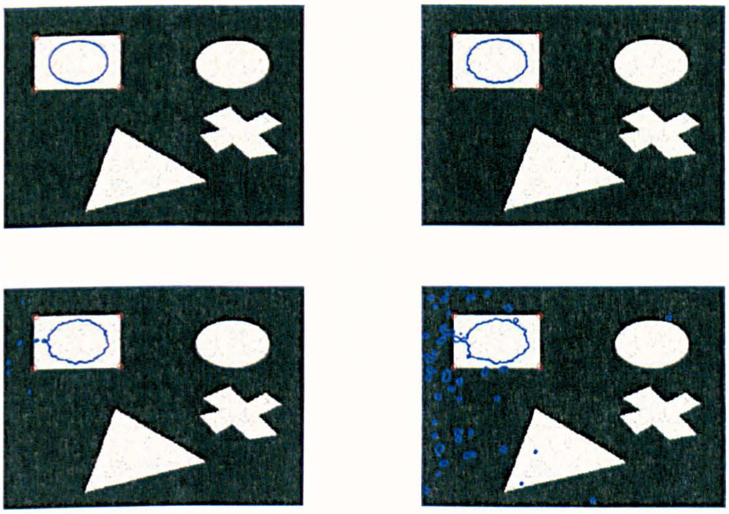


Figure 7.2: Results **M-1** tested on synthetic noisy image with 4 markers and initial guess $\phi_0 = \sqrt{(x-x_0)^2+(y-y_0)^2} - 25$, where x_0, y_0 are the averages of x, y-components of the markers, $\alpha = -0.00151$ (different α have been used) and $\sigma = 4$. Top Left: Original image with initial data. Top Right: Result after 1000 iterations. Bottom Left: Result after 6000 iterations. Bottom Right: Result after 20000 iterations.

In figure 7.3, **M-1** is tested on synthetic noisy image, where the image is filtered first. The initial condition is $\phi_0 = \sqrt{(x-x_0)^2+(y-y_0)^2} - 25$, where x_0, y_0 are the averages of x, y-components of the markers. Top left figure is the original image with initial contour and the top right figure is the result after 800 iterations. Bottom left figure is the result after 6000 iterations and bottom right figure is the final result after 16000 iterations. The final result is not very satisfactory.

In figure 7.4, **M-1** is tested on an artificial image. Here it is only able to detect the outer boundary of the letter O.

In figure 7.5 our model **M-2** is tested on an artificial image to detect the object X with 3 markers and initial condition is $\phi_0 = \sqrt{(x-x_0)^2+(y-y_0)^2} - r_0$, where x_0, y_0 are defined as above and $r_0 = \min_{\mathbf{y}} \|\mathbf{x} - \mathbf{y}\|$ where $\mathbf{x} = (x_0, y_0)$ and $\mathbf{y} \in A$. Top left image is the original image with initial data and top right figure is the result after 1 iteration. Bottom left figure is the result after 4 iterations and bottom right figure is the final result in 14 iterations. The object X is segmented successfully.

In figure 7.6, **M-2** is tested on synthetic noisy image with 4 markers and initial guess is $\phi_0 = \sqrt{(x-x_0)^2+(y-y_0)^2} - 25$ with parameters $\mu = 10$, $\alpha = -0.01$, $\lambda_1 = 0.01$, $\lambda_2 = 0.01$ and $\sigma = 4$. Top left figure is the original image with initial data and top right is the result after 15 iteration. Bottom left is the result after 30 iterations and bottom right figure is the final result after 130 iterations, the required object is successfully detected without filtering the noise (i.e we use the original image with noise).

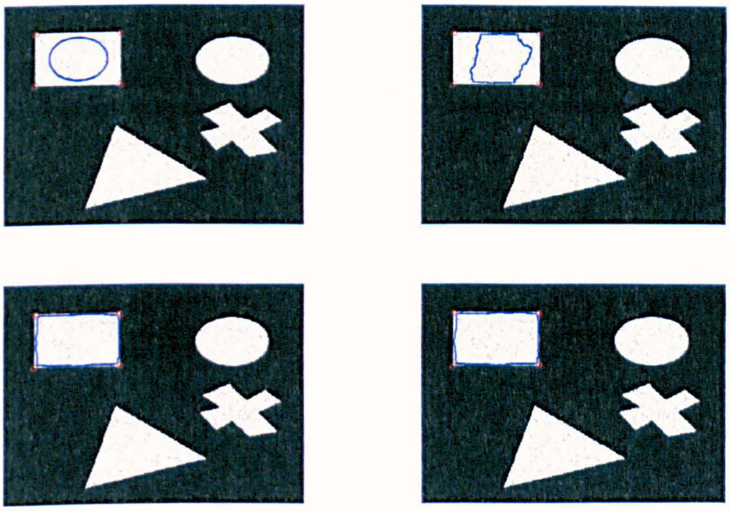


Figure 7.3: Results of **M-1** tested on filtered ($z * K_\sigma$, where K_σ may be Laplace or Gaussian filter) synthetic noisy image with 4 markers with initial guess $\phi_0 = \sqrt{(x-x_0)^2+(y-y_0)^2} - 25$, where x_0, y_0 are the averages of x, y-components of the markers, $\alpha = -0.0011$ and $\sigma = 4$. Top Left: Original image with initial data. Top Right: Result after 800 iterations. Bottom Left: Result after 6000 iterations. Bottom Right: Result after 16000 iterations.

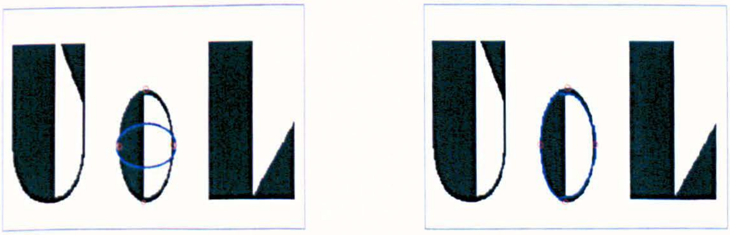


Figure 7.4: Experimental results of **M-1** on an artificial image. Only able to detect the outer boundary and unable to find the inner boundary of the letter O. Left: Original image with initial data. Right: Final result after 1000 iterations.

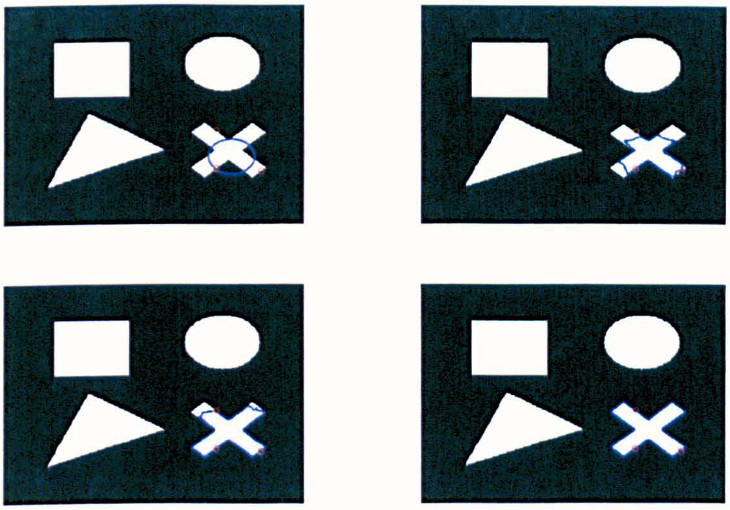


Figure 7.5: We first test **M-2** on an artificial image. Our new model is used to detect the object X in an artificial image with 3 markers with initial guess $\phi_0 = \sqrt{(x-x_0)^2+(y-y_0)^2}-r_0$, x_0, y_0 are the averages of the x, y-components of the markers and $r_0 = \min_{\mathbf{y}} \|\mathbf{x}-\mathbf{y}\|$ where $\mathbf{x} = (x_0, y_0)$ and $\mathbf{y} \in A$, $\mu = 10$, $\alpha = -0.01$ and $\sigma = 4$. Top Left: Original image with initial data. Top Right: Result after 1 iteration. Bottom Left: Result after 4 iterations. Bottom Right: Result after 14 iterations.

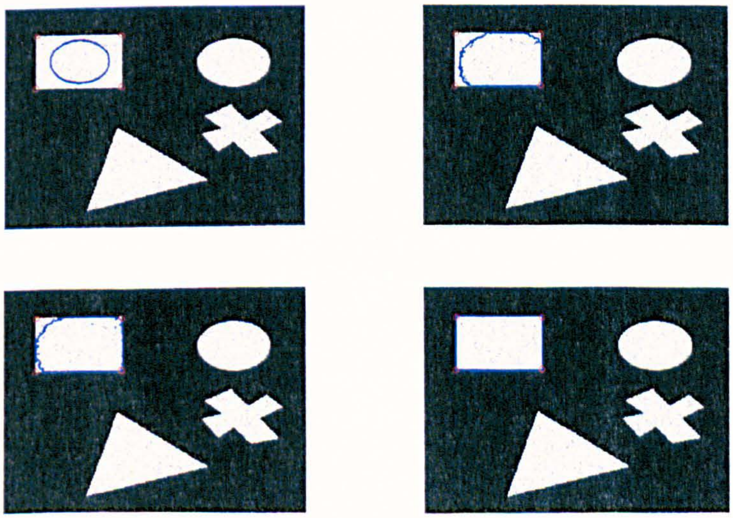


Figure 7.6: To detect the rectangle in noisy image with 4 markers with initial guess $\phi_0 = \sqrt{(x-x_0)^2+(y-y_0)^2} - 25$ where x_0, y_0 are same as defined above, $\mu = 10$, $\alpha = -0.01, \lambda_1 = 0.01, \lambda_2 = 0.01$ and $\sigma = 4$.

In figure 7.7 **M-2** is tested on an artificial image and segmented the image successfully **M-1** did not do very well as shown in figure 7.4. Initial condition is $\phi_0 = \sqrt{(x-x_0)^2+(y-y_0)^2} - r_0$, where x_0 and y_0 are the average of x, y-components of the markers respectively and r_0 is the same as defined above. Other parameters are μ = (size of $z)^2/1400$,$\lambda_1 = 0.00951, \lambda_2 = 0.0095$, $\alpha = -5.1 \times 10^{-4}$ and $\sigma = 4$. Top left figure is the original image with initial data and the top right is the result after 35 iterations. Bottom left is the result after 90 iterations and bottom right is the final result after 250 iterations. In figure 7.8, it is shown that if the markers are not exactly on boundary, the object can be detected. The same data is used as in the above example.

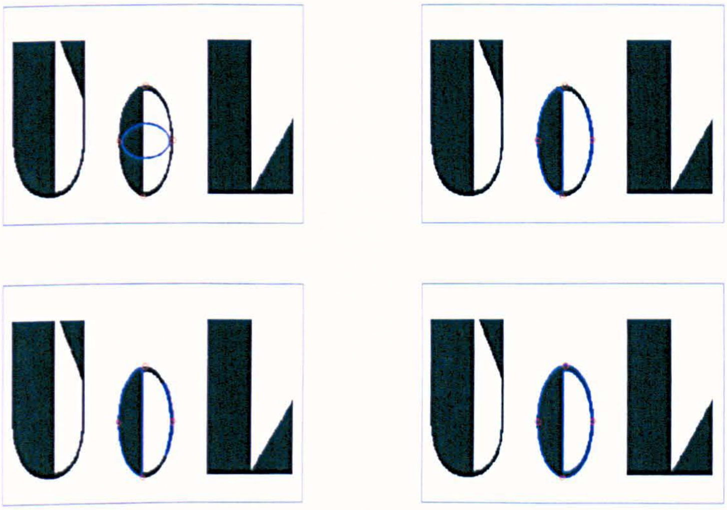


Figure 7.7: To detect the letter O in the image UOL with 4 markers with initial guess $\phi_0 = \sqrt{(x-x_0)^2+(y-y_0)^2} - r_0$, where x_0 and y_0 are the average of x, y-components of the markers respectively. μ = (size of $z)^2/1400$,$\lambda_1 = 0.00951, \lambda_2 = 0.0095$, $\alpha = -5.1 \times 10^{-4}$ and $\sigma = 4$.

In figure 7.9, we show results where **M-1, M-2** fail to segment the image if the initial guess ϕ_0 is far from the markers (away from the object to be detected). In figure 7.10 our model is tested on a real brain MRI image to detect a tumor with 4 markers. The initial condition is $\phi_0 = \sqrt{(x-x_0)^2+(y-y_0)^2} - r_0$, where x_0 and y_0 are the average of x, y-components of the markers respectively. The other parameters used are μ = (size of $z)^2/10$,$\lambda_1 = 0.0001, \lambda_2 = 0.0001$, $\alpha = -1.51 \times 10^{-2}$ and $\sigma = 4$. Top left figure is the original image with initial data and top right figure is the result after 10 iterations. Bottom left figure is the result after 40 iterations and bottom right figure is the final result after 200 iterations.

In figure 7.11, we test the model on real knee MRI image with 3 markers (also work with 2 markers), the initial condition is $\phi_0 = \sqrt{(x-x_0)^2+(y-y_0)^2} - r_0$, where x_0 and y_0 are the average of x, y-components of the markers respectively with the following parameters μ = (size of $z)^2/10$,$\lambda_1 = 0.000051, \lambda_2 = 0.000051$, $\alpha = -1.51 \times 10^{-3}$ and $\sigma = 4$. Top left figure is the original image with initial data and top right figure is the result after 20 iterations. Bottom left figure is the result after 40 iterations and bottom right figure is the final result after 120 iterations.

Lastly, we compare results by number of iterations. In figure 7.12, results obtained from applying M-1 method on an artificial image to detect the disc in the image, where

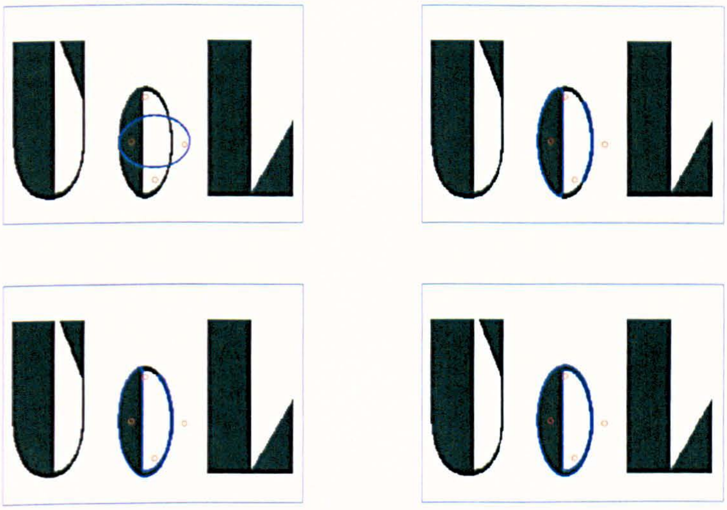


Figure 7.8: To detect the letter O in the image UOL with 4 markers placed away from the boundaries with initial guess $\phi_0 = \sqrt{(x - x_0)^2 + (y - y_0)^2} - 20$, where x_0 and y_0 are the average of x, y-components of the markers respectively. $\mu = (\text{size of } z)^2/1400, \lambda_1 = 0.00951, \lambda_2 = 0.0095$, $\alpha = -5.1 \times 10^{-4}$ and $\sigma = 4$.

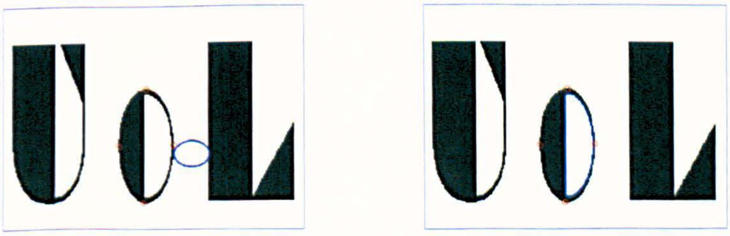


Figure 7.9: ϕ_0 should be inside the markers. Results if the initial guess is outside the markers.

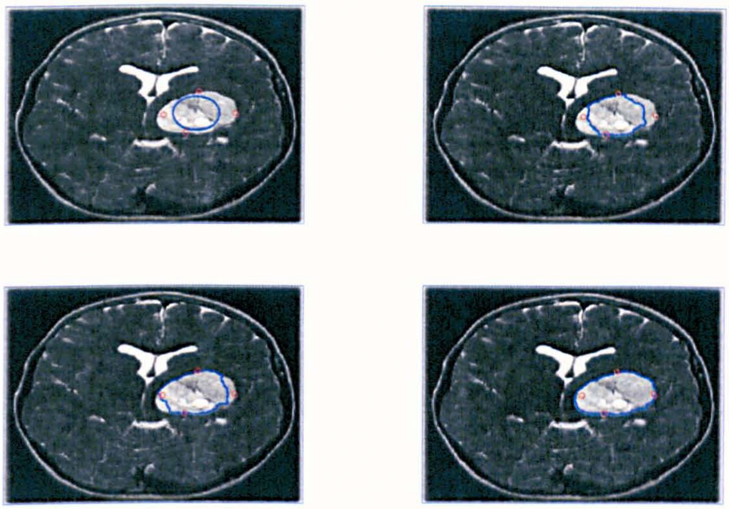


Figure 7.10: To detect a tumor in a real brain MRI image with 4 markers with initial guess $\phi_0 = \sqrt{(x-x_0)^2+(y-y_0)^2} - r_0$, where x_0 and y_0 are the average of x, y-components of the markers respectively. $\mu =$ (size of $z)^2/10$,$\lambda_1 = 0.0001$, $\lambda_2 = 0.0001$, $\alpha = -1.51 \times 10^{-2}$ and $\sigma = 4$. This task can be done using geodesic active contours.

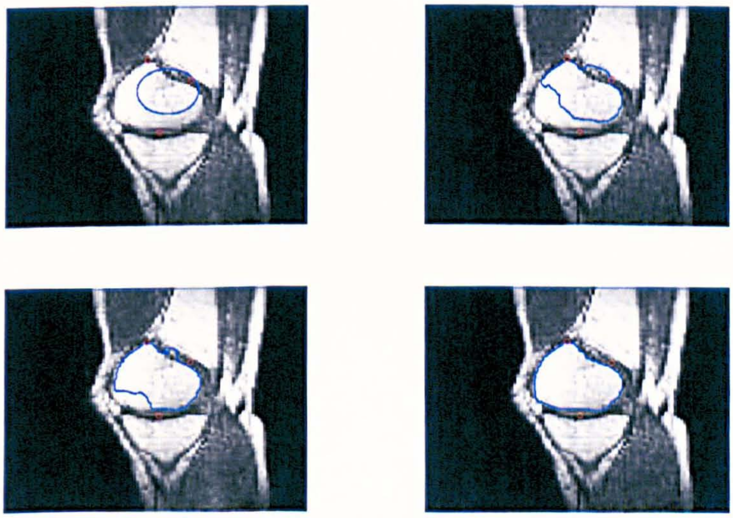


Figure 7.11: A real knee MRI image with 3 markers with initial guess $\phi_0 = \sqrt{(x-x_0)^2+(y-y_0)^2} - r_0$, where x_0 and y_0 are the average of x, y-components of the markers respectively. $\mu =$ (size of $z)^2/10$,$\lambda_1 = 0.000051$, $\lambda_2 = 0.000051$, $\alpha = -1.51\times10^{-3}$ and $\sigma = 4$.

the boundary is not defined by the gradient, the disc is successfully detected in 500 iterations. In figure 7.13, our model M-2 is tested on the same image and the final result is obtained in 100 iterations.

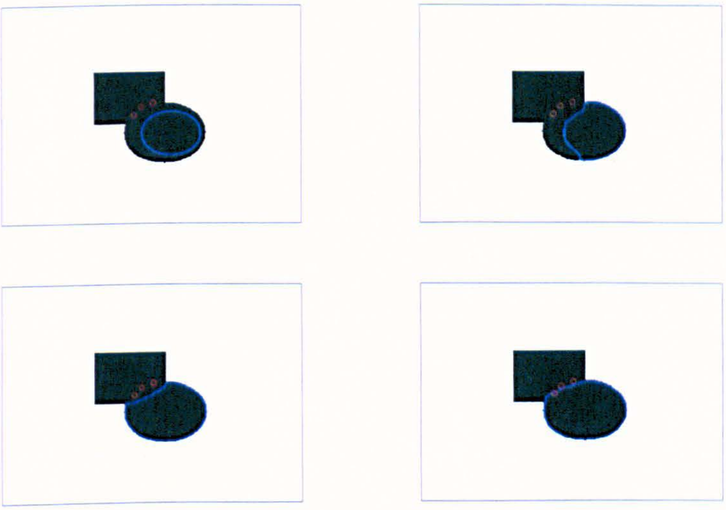


Figure 7.12: To detect the disc in the disc-rectangle image using **M-1**, the disc and rectangle have the same intensity and the boundary can not be defined by gradient. Using 3 markers with initial guess $\phi_0 = \sqrt{(x-150)^2 + (y-145)^2} - 25$, $\alpha = -0.025$ and $\sigma = 4$. Top Left: Original image with initial data. Top Right: Result after 20 iterations. Bottom Left: Result after 100 iterations. Bottom Right: Result after 500 iterations.

7.6 Conclusion

In this chapter we presented a new model based on geodesic active contours and the Chan-Vese model. Our proposed model is good for noisy images without Gaussian filter. We also tested the new model on real images. Our model is also faster then the existing model of Gout et al in term of number of iterations. The model works if the markers are not on the desired boundary i.e we can detect an object if the markers are apart from its boundary. This model is dependent on the initial guess- the initial guess should be inside the markers which is shown in the numerical experiments.

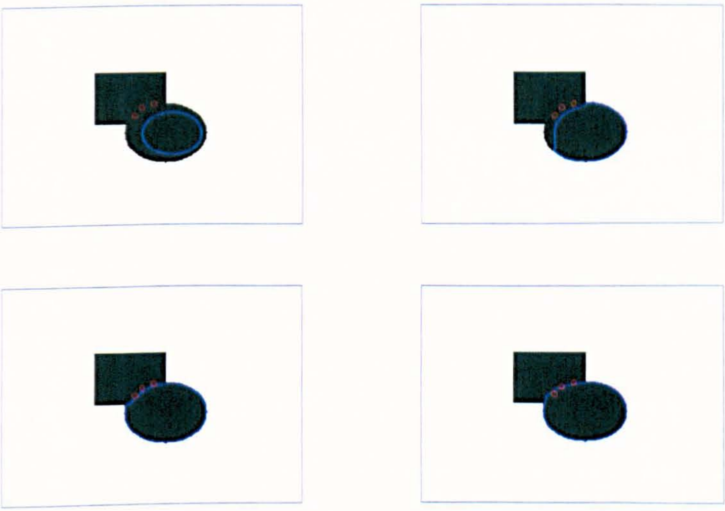


Figure 7.13: To detect the disc in the disc-rectangle image using our model **M-2**, the disc and rectangle have the same intensity and the boundary can not be defined by gradient. Using 3 markers with initial guess $\phi_0 = \sqrt{(x-150)^2 + (y-145)^2} - 25$, $\mu =$ size of image$^2/5000$, $\alpha = -0.1$ and $\sigma = 4$. Top Left: Original image with initial data. Top Right: Result after 5 iterations. Bottom Left: Result after 15 iterations. Bottom Right: Result after 100 iterations.

Chapter 8

Conclusion and Further Directions

This thesis presented four new algorithms for solving image segmentation problems. All are mainly based on the Chan-Vese segmentation model (active contour without edges).

8.1 Achievements

The first model discussed in this thesis is the Chan-Vese model [35] "Active Contour without Edges", a special case of piecewise constant Mumford and Shah model [88] with level set formulation (2 phase segmentation). A non linear multigrid method was developed for solving the partial differential equation arising from the minimization of the above mentioned Chan-Vese model [35]. Then the main advantage of this method is the speed of convergence in terms of CPU time. Moreover it was found that multigrid method with the smoother discussed in this thesis can get improved solution for global minimum of the functional. And this is shown by giving some experimental results. The effectiveness of smoother is checked by using local Fourier analysis. Smoothing analysis of the two different smoothers (local and global smoother) is also presented in this thesis.

Secondly, we have discussed multiphase image segmentation model [120]. A non linear multigrid method, developed for 2 phase image segmentation, was generalized to multiphase image segmentation model, but the results were not very good, which was observed from the smoothing analysis of the smoother. A new modified smoother is introduced which improved the results improvement in sense of quality of the segmented image and CPU time. Nevertheless multiphase image segmentation model [120] is dependent on initial condition. To avoid this problem we proposed an algorithm which implement 2 phase multigrid algorithm in hierarchical way to get multiphase image segmentation results. We presented the comparison of different methods.

Thirdly, we developed a new optimization based multilevel technique [30] for the 2 phase image segmentation model [35]. We have described a model proposed by Chan et al [25] and Bresson et al [15] for finding global minimum of the CV model. This new multilevel algorithm allowed us to reach global minimum of the original CV functional by comparing the minimum values with the values obtained from the functional mentioned above. We have also applied this technique to the global minimization model for CV model to improve the results. Minimum values and experimental results are presented for comparison.

Finally, we proposed a new model for image segmentation under geometrical conditions like set of points. In this model we combined geodesic contour model with CV model, which allowed the new model to be useful on noisy input images. As expected, the new model can detect objects of interest in very noisy images. This model is also very fast in implementation by the number of iterations. For solving the PDE arising from the minimization of the new model additive operator splitting (AOS) method is applied.

8.2 Future Work

- We will develop a multigrid algorithm and an optimization based multilevel method for 2 phase piecewise smooth approximation of Mumford and Shah model. Such a model is more general than CV [35].
- We will develop a multigrid algorithm and an optimization base multilevel method for the model developed for image segmentation under geometrical conditions. Selective segmentation work is new to the literature. There are few published results.
- We will work on joint image segmentation and image registration. This will save processing time for video images.
- Of course, it is of interest to consider 3D segmentation models and fast algorithms..

Bibliography

[1] S.T. Acton. Multigrid anisotropic diffusion. *T-IP*, 7:280–291, 1998.

[2] R. Adams and L. Bischof. Seeded region growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6), 1994.

[3] L. Alvarez, P.-L Lions, and J. M. Morel. Image selective smoothing and edge detection by nonlinear diffusion. *SIAM J. Numerical Analysis.*, 29(3):845–866, 1992.

[4] L. Ambrosio and V. M. Tortorelli. Approximation of functionals depending on jumps by elliptic functionals via γ-convergence. *Comm. Pure Appl. Math.*, 43:999–1036, 1990.

[5] D. Apprato, D. Ducassou, C. Gout, E. Laffon, and C. L. Guyader. Segmentation of medical image sequence under constraints: application to non-invasive assessment of pulmonary arterial hypertension. *International Journal of Computer Mathematics*, 81(5):527–536, 2004.

[6] T. Asano, D. Z. Chen, N. Katoh, and T. Tokuyama. Polynomial-time solutions to image segmentation. *Proc. of the 7th Ann. SIAM-ACM Conference on Discrete Algorithms*, 1996.

[7] G. Aubert and P. Kornprobst. *Mathematical problems in image processing: Partial Differential Equations and the Calculus of Variations.* Springer, 2002.

[8] N. Badshah and K. Chen. Multigrid method for the chan-vese model in variational segmentation. *Comm. Comput. Phys.*, 4(2):294–316, 2008.

[9] N. Badshah and K. Chen. On two multigrid algorithms for modeling variational multiphase image segmentation. *IEEE Transactions on Image Processing, to appear*, 2008.

[10] N. Badshah and Ke Chen. An optimization-based multilevel method for variational image segmentation models. *Electronic Transactions on Numerical Analysis* , Submitted, 2008.

[11] R. P. Beyer and R. J. Leveque. Analysis of a one-dimensional model for the immersed boundary method. *SIAM Journal of Numerical Analysis*, 29(2):332–364, 1992.

[12] P. Blomgren, T.F Chan, P. Mulet, L. A. Vese, and W. L. Wan. Variational PDE models and methods for image processing. *Research Notes in Mathematics.*, 420:43–67, 2000.

[13] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization.* Cambridge University Press The Edinburgh Building, Cambridge, CB2 8RU, UK, 2004.

[14] Achi Brandt. Multi-level adaptive solutions to boundary-value problems. *Math. Comp.*, 31(138):333–390.

[15] X. Bresson, S. Esedoglu, P. Vandergheynst, J.P Thiran, and S. Osher. Global minimizers of the active contour/snake model. *CAM Report 05-04*, 2005.

[16] William L. Briggs. A multigrid tutorial.

[17] Carlos Brito-Loeza and Ke Chen. Multigrid method for a modified curvature driven diffusion model for image inpainting. *Journal of Computational Mathematics*, 26(6):856–875, 2008.

[18] LI Can-Fei, WANG Yao-Nan, and LIU Guo-Cai. A new splitting active contour framework based on chan-vese piecewise smooth model. *Acta Automatica Sinica*, 34(6):659–664, 2008.

[19] V. Caselles. Geometric models for active contours. *ICIP 95*, pages 9–12, 1995.

[20] V. Caselles, Francine Catté, Tomeu Coll, and Françoise Dibos. A geometric model for active contours in image processing. *Numer. Math*, 66(1):1–31, 1993.

[21] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contours. *International Journal of Computer Vision*, 22(1):61–79, 1997.

[22] S. Chabrier, C. Rosenberger, B. Emile, and H. Laurent. Optimization-based image segmentation by genetic algorithms. *EURASIP Journal on Image and Video Processing*, 2008, 2008.

[23] Antonin Chambolle. Image segmentation by variational methods: Mumford and shah functional and the discrete approximations. *SIAM J. Appl. Math.*, 55(3):827–863, 1995.

[24] T. F. Chan, K. Chen, and Tai X. C. *Nonlinear multilevel schemes for solving the total variation image minimization problem.* Second edition. Interscience Tracts in Pure and Applied Mathematics, No. 4. Springer Berlin Heidelberg, 2007.

[25] T. F. Chan, S. Esedoglu, and M. Nikolov. Algorithm for finding global minimizers of image segmentation and denoising models. *UCLA CAM Report*, 04-54, 2004.

[26] T. F. Chan, G.H. Golub, and P. Mulet. A nonlinear primal dual method for total variation based image restoration. *SIAM J. Sci. Comp.*, 20(6):1964-1977, 1999.

[27] T. F. Chan and L. A. Vese. An efficient variational multiphase motion for the mumford-shah segmentation model. *in proceedings of the 34th Asilomar Conference on Signals, Systems and Computers (ACSSC '00)*, 1:490–494, 2000.

[28] T. F. Chan and W. L. Wan. Robust multigrid methods for non-smooth coefficient elliptic linear systems. *Journal of Computational and Applied Mathematics*, 123:323–352, 2000.

[29] Tony F. Chan and Ke Chen. On a nonlinear multigrid algorithm with primal relaxation for the image total variation minimization. *Numerical Algorithms*, 41:387–411, 2006.

[30] Tony F. Chan and Ke Chen. An optimization based multilevel agorithm for total variation image denoising. *SIAM J. Multiscale Modeling Simulation (MMS)*, 5(2):615–645, 2006.

[31] Tony F. Chan, Ke Chen, and Jamylle L. Carter. Iterative methods for solving the dual formulation arising from image restoration. *Electronic Transactions on Numerical Analysis*, 26:299-311, 2007.

[32] Tony F. Chan, Yezrielev B. Sandberg, and Luminita A. Vese. Active contours without edges for vector-valued images. *Journal of Visual Communication and Image Representation*, 11(2):130–141, 2000.

[33] Tony F. Chan and Jianhong Shen. *Image Processing and Analysis.* Society for Industrial and Applied Mathematics SIAM, 3600 University City Science Center, Philadelphia, PA 19104-2688, USA, 2005.

[34] Tony F. Chan, Jianhong Shen, and Luminita Vese. Variational pde models in image processing. *Notices Am. Math. Soc.*, 50(1):14-26, 2003.

[35] Tony F. Chan and Luminita A. Vese. Active contours without edges. *IEEE Transactions on Image Processing*, 10(2):266–277, 2001.

[36] Tony F. Chan and Lumunita A. Vese. An efficient variational multiphase motion for the mumford-shah segmentation model. *IEEE Asilomar Conference on Signals Systems and Computers.*, 1:490–494, 2000.

[37] Q. S. Chang and I. Chern. Acceleration methods for total variation based image denoising. *SIAM J. Sci. Comput.*, 25:982–994, 2003.

[38] K. Chen and J. Savage. An accelerated algebraic multigrid algorithm for total variation denoising. *BIT*, 47:277–296, 2007.

[39] Ke Chen. *Matrix Preconditioning Techniques and Applications.* Cambridge University Press, The Edinburgh Building, Cambridge CB2 2RU,UK, first edition, 2005.

[40] Ke Chen and Xue-Cheng Tai. A nonlinear multigrid method for total variation minimization from image restoration. *Journal of Scientific Computing*, 41(2):115–138, 2007.

[41] Li Chen, Yue Zhou, Yonggang Wang, and Jie Yang. Gacv: Geodesic-aided CV method. *Pattern Recognition*, 39:1391-1395, 2006.

[42] Y. G. Chen, Y. Giga, and S. Goto. Uniqueness and existence of viscosity solutions of generalized mean curvature flow equations. *Journal of Differential Geometry*, 33(3):749-786, 1991.

[43] Noppadol Chumchob and Ke Chen. A robust affine image registration method. *International Journal of Numerical Analysis and Modeling*, 6(3):311–334, 2009.

[44] Ginmo Chung and Luminita A. Vese. Image segmentation using a multilayer levelset approach, Regular Article. *Computing and Visualization in Science*, 2008.

[45] D. L. Cohen. On active contour models and ballons. *CVGIP: Image Understanding*, 53:211-218, 1991.

[46] L. D. Cohen. On active contours models and balloons. *Computer Vision, graphics and image processing: Image Understanding*, 53(2):211–218, 1991.

[47] Samuel Daniel Conte and Carl W. De Boor. *Elementary Numerical Analysis: An Algorithmic Approach*. McGraw-Hill Higher Education, 1980.

[48] P. M. De Zeeuw. Matrix dependent prolongations and restriction in a blackbox Multigrid Solver. *Journal of Computational and Applied Mathematics*, 33:1–27, 1990.

[49] J. E. Dendy. Black Box Multigrid. *Journal of Computational Physics*, 48:366–386, 1982.

[50] Xaiojun Du and Tien D. Bui. A new model for image segmentation. *IEEE Signal Processing Letters*, 15:182–185, 2008.

[51] H. Engl, M. Hanke, and A. Neubauer. *Regularization of Inverse Problems*. Kluwer Academic Publishers, 1996.

[52] S. Esedoglu and Y. H. Tsai. Threshold dynamics for the piecewise constant mumford-shah functional. *J. Comput. Phys*, 211(1), 2006.

[53] L. C. Evan and J. Spruck. Motion of level sets by mean curvature i. *Journal of Differential Geometry*, 33(3):635–681, 1991.

[54] L. C. Evan and J. Spruck. Motion of level sets by mean curvature ii. *Transaction of the American Mathematical Society*, 330:321–332, 1992.

[55] L. C. Evan and J. Spruck. Motion of level sets by mean curvature iii. *Journal of Geometric Analysis*, pages 121–150, 1992.

[56] J. Douglas Faires and Richard Burden. *Numerical methods*. Brooks/Cole Publishing Co., Pacific Grove, CA, sixth edition, 1998.

[57] F. Gibou and R. Fedkiw. A fast hybrid k-mean level set algorithm for segmentation. *4th Annual Hawaii International Conference on Statistics and Mathematics*, pages 281–291, 2005.

[58] De E. Giorgi, M. Carriero, and A. Leaci. Existence theorem for a minimum problem with free discontinuity set. *Arch. Rational Mech. Anal.*, 108(3):195–218, 1989.

[59] Enrico Giusti. *Minimal Surfaces and Functions of Bounded Variation*. Birkhuser Boston, 1984.

[60] R. Goldenberg, R. Kimmel, E. Rivlin, and M. Rudzsky. Fast geodesic active contours. *IEEE Transaction on image processing*, 10(10):1467–1475, 2001.

[61] C Gout, C. Le Guyader, and Luminita Vese. Segmentation under geometrical consitions with geodesic active contour and interpolation using level set methods. *Numerical Algorithms*, 39:155–173, 2005.

[62] Christian Gout and Carole Le Guyader. Segmentation of complex geophysical structures with well data. *Computational Geosciences*, 10(4):361–372, 2006.

[63] Christian Gout and Sylvie Vieira-Test. An algorithm for segmentation under interpolation conditions using deformable models. *International Journal of Computer Mathematics*, 80(1):47–54, 2003.

[64] C. W. Groetsch. *The Theory of Tikhonov Regularization for Fredholm Equations of the First Kind (Research Notes in Mathematics Series).* Longman Sc & Tech, 1984.

[65] C. Le Guyader and C Gout. Geodesic active contour under geometrical conditions theory and 3d applications. *Numerical Algorithms*, 48:105–133, 2008.

[66] Carole Le Guyader, Dominique Apprato, and Christian Gout. Using a level set approach for image segmentation under interpolation conditions. *Numerical Algorithms*, 39(1-3):221–235, 2005.

[67] Carole Le Guyader and L. A. Vese. Self-repelling snakes for topology-preserving segmentation models. *IEEE Transaction on Image Processing*, 17(5):767–779, 2008.

[68] V. E. Henson. Multigrid methods nonlinear problems: an overview. *Computational Imaging*, 5016:36–48, 2003.

[69] W. Hinterberger, Hintermüller M., Kunisch K., Markus Von Oehsen, and Otmar Scherzer. Tube methods for bv regularization. *J. Math. Imaging Vis.*, 19(3):219–235, 2003.

[70] M. Hintermüller and Kunisch K. Total bounded variation regularization as a bilaterally constrained optimization problem. *SIAM J. Appl. Math.*, 64(4):1311–1333, 2004.

[71] M. Hintermüller and G. Stadler. An infeasible primal-dual algorithm for total bounded variation–based inf-convolution-type image restoration. *SIAM J. Sci. Comput.*, 28(1):1–23, 2006.

[72] M. Jeon, M. Alexander, W. Pedrycz, and N. Pizzi. Unsupervised hierarchical image segmentation with level set and additive operator splitting. 26(10):1461–1469, July 2005.

[73] G. S. Jiang and D. Peng. Weighted eno schemes for hamilton-jacobi equations. *Journal of Scientific Computing*, 21(6):2126–2143, 1999.

[74] Yoon Mo Jung, Sung Ha Kang, and Jianhong Shen. Multiphase image segmentation via modica-mortola phase transition. *SIAM Journal of Applied Mathematics*, 67(5):1213–1232, 2007.

[75] M. Kass, A. Witkin, and A. Terzzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 6(4):321–331, 1987.

[76] Y. Leclerc. Region growing using the MDL principle. In DAPPRA image understanding workshop, 1990.

[77] Chunming Li, Chenyang Xu, Changfeng Gui, and M.D. Fox. Level set evolution without re-initialization: a new variational formulation. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 1:430–436, June 2005.

[78] H. W. Li and Xue-Cheng Tai. Piecewise constant level set methods for multiphase motion. *Int. J. Numer. Anal. Modelling*, 4(2):291–305, 2007.

[79] J. Lie, M. Lysaker, and X.-C. Tai. A variant of the level set method and applications to image segmentation. *Mathematics of Computation*, 75:1155–1174, 2006.

[80] F. Liu, Y. Luo, and D. Hu. Adaptive level set image segmentation using the Mumford and Shah functional. *Optical Engineering*, 41(12):3002–3003, 2002.

[81] T. Lu, P. Neittaanmaki, and X.-C. Tai. A parallel splitting up method for partial differential equations and its application to navier-stokes equations. *RAIRO Math. Model. and Numerical Analysis*, 26(6):673–708, 1992.

[82] S. Mahmoodi and B. S. Sharif. Contour evolution scheme for variational image segmentation and smoothing. *Image Processing, IET*, 1(3):287–294, September 2007.

[83] R. Malladi, J. Sethian, and B. Vemuri. Shape modeling with front propagation: A level set approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:158–175, 1995.

[84] John Miller and C. John Bancroft. Multigrid priciples. *CREWES Research Report*, 15, 2003.

[85] L. Modica. The gradient theory of phase transitions and the minimal interface criterion. *Arch. Rational Mech. Anal.*, 98:123–142, 1987.

[86] J. M. Morel and S. Solimini. Variational methods in image segmentation: A constructive approach. *Revista Matematica Universidad Complutense de Madrid*, 1:169–182, 1988.

[87] J. M. Morel and S. Solimini. *Variational methods in image segmentation.* Birkhauser Boston Inc., Cambridge, MA, USA, 1995.

[88] David Mumford and Jyant Shah. Optimal approximation by piecewise smooth functions and associated variational problems. *Communications on Pure Applied Mathematics*, 42:577–685, 1989.

[89] Markus Von Oehsen. *Multiscale Methods for Variational Image Processing.* Logos Verlag Berlin, Comeniushof, Gubener str. 47, 10243 Berlin, first edition, 2002.

[90] S. Osher and R. Fedkiw. *Level Set Methods and Dynamic Implicit Surfaces.* Springer, 2003.

[91] S. Osher and C. W Shu. High order essentially non-oscillatory schemes for hamilton-jacobi equaitons. *Journal of Computational Physics*, 28(4):907–922, 1991.

[92] Stanley Osher and James A. Sethian. Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations. *J. Comput. Phys.*, 79(1):12–49, 1988.

[93] Yongsheng Pan, J. Douglas Birdwell, and Seddik M. Djouadi. Efficient implementation of the chan-vese models without solving pdes. *Multimedia Signal Processing, 2006 IEEE 8th Workshop*, pages 350–354, Oct. 2006.

[94] G. Papandreou and P. Maragos. A fast multigrid implicit algorithm for the evolution of geodesic active contours. In *Proc. IEEE Int. Conf. on Comp. Vision and Pat. Rec. (CVPR-04), Washington DC, June 2004*, volume II, pages 689–694, 2004.

[95] G. Papandreou and P. Maragos. Multigrid geometric active contour models. *IEEE Transactions on Image Processing*, 16(1):229–240, 2007.

[96] N. Paragios and R. Deriche. Geodesic active contours and level sets for the detection and tracking of moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(3):266–280, 2000.

[97] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C, The art of Scientific Computing.* Cambridge University Press, 1992.

[98] Robert D. Richtmyer and K. W. Morton. *Difference methods for initial-value problems.* Second edition. Interscience Tracts in Pure and Applied Mathematics, No. 4. Interscience Publishers John Wiley & Sons, Inc., New York-London-Sydney, 1967.

[99] Leonid I. Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithm. *Physica D North Holland*, 60(1-4):259–268, 1992.

[100] M. Rudzsky, E. Rivlin, R. Kimmel, and R. Goldenberg. Fast geodesic active contours. *Scale-Space Theories in Computer Vision*, 1682:34–45, 1999.

[101] J. W Ruge and K. Stuben. Algebraic multigrid, in multigrid methods, Frontiers in Applied Mathematics. *SIAM, Philadelphia*, 3:73–130, 1987.

[102] Yezrielev B. Sandberg, Tony F. Chan, and Luminita A. Vese. A Level-Set and Gabor-Based Active Contour Algorithm for Segmenting Textured Images. *UCLA CAM Report.*, 02–39, 2002.

[103] G. Sapiro, R. Kimmel, and V. Caselles. Geodesic active contours. *ICCV*, 95:694–699, 1995.

[104] Guillermo Sapiro. *Geometric partial differential equations and image analysis.* Cambridge University Press, New York, NY, USA, 2001.

[105] Joseph Savage and Ke Chen. An improved and accelerated non-linear multigrid method for total-variation denoising. *International Journal of Computer Mathematics*, 82(8):1001–1015, August 2005.

[106] Joseph David Savage. *Fast Iterative Methods for Solving Large Systems Arising from Variational Models in Image Processing.* PhD thesis, 2006.

[107] Christoph Schnorr. A study of a convex variational diffusion approach for image segmentation and feature extraction. *Journal of Mathematical Imaging and Vision*, 8:271-292, 1998.

[108] J. A. Sethian. *Level Set Methods and Fast Marching Methods:* Evolving interfaces in computational Geometry, Fluid Mechanics, Computer Vision, and Materials Sciences. Cambridge University Press, The Edinburgh Building,Cambridge, UK, 2002.

[109] Jianhong H. Shen. Gamma-convergence approximation to piecewise constant mumford-shah segmentation, J. Blanc-Talon et al. *acivs 2005 (int'l conf. advanced concepts intell. vision systems). Lec. Notes Comp. Sci.*, 3708, 2005.

[110] Bing Song and Tony F. Chan. A fast algorithm for level set based optimization. *CAM-UCLA*, 02–68, 2002.

[111] M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis, and Machine Vision.* Chapman & Hall, 2 edition, 1998.

[112] D. M. Strong and T. F Chan. Exact solutions to total variation regularization problems. *UCLA CAM Report 96-41*, 1996.

[113] D. M. Strong and Tony F. Chan. Edge preserving and scale dependant properties of total variation regularization. *Inverse Problems*, 19:165–187, 2003.

[114] M. Sussman, P. Smereka, and S. Osher. A level set approach for computing solutions to incompressible two-phase flow. *Journal of Computational Physics*, 114:146–159, 1994.

[115] X.-C. Tai, K.-A. Lie, T.F. Chan, and S. Osher, editors. *Image Processing Based on Partial Differential Equations: Proceedings of the International Conference on PDE-Based Image Processing and Related Inverse Problems.* Springer, 2005.

[116] I. Tamanini and G. Congedo. Optimal segmentation of unbounded functions. *Rend. Sem. Mat. Univ. Padova*, 95:153–174, 1996.

[117] A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill Posed Problems.* V. H. Winston, 1977.

[118] Ulrich Trottenberg and Anton Schuller. *Multigrid.* Academic Press, Inc., Orlando, FL, USA, 2001.

[119] L. A. Vese. Multiphase object detection and image segmentation. *UCLA CAM Report*, 02-36, 2002.

[120] Luminita A. Vese and Tony F. Chan. A multiphase level set framework for image segmentation using the mumford and shah model. *Int. Journal of Computer Vision*, 50:271–293, 2002.

[121] L. Vincent and Soille. Watersheds in digital spaces - an efficient algorithm based on immersion. *IEEE Trans. Pattern Analysis and Machine Learning.*, 6:583–598, 1994.

[122] Curtis R. Vogel. *Computational Methods for Inverse Problems.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2002.

[123] Curtis R. Vogel and M. E. Oman. Iterative methods for total variation denoising. *SIAM Journal of Sci. Statis. Comput.*, 17:227–238, 1996.

[124] W. L. Wan, T. F. Chan, and B. Smith. An energy-minimizing interpolation for robust multigrid methods. *SIAM J. Scientific Computing*, 21:1632–1649, 2000.

[125] Wenyuan WANG, Haibo Xu, and Suhua Wei. An active contour model for selective segmentation. *Acta Automatica Sinica*, 2(9):36–49, 2005.

[126] J. Weickert and G. Kühne. Fast methods for implicit active contours models, preprint 61. *Universität des Saarlandes, Saarbrücken*, 2002.

[127] Joachim Weickert, Bart M. ter Haar Romeny, and Max A. Viergever. Efficient and reliable schemes for nonlinear diffusion filtering. *Scale-Space theory in computer vision, Lecture Notes in Computer Science*, 1252:260–271, 1997.

[128] P. Wesseling. *An Introduction to Multigrid Methods.* John Wiley & Sons, 1992.

[129] C. Y. Xu and J. L. Prince. Snakes, shapes and gradient vector flow. *IEEE Trans. Image Proc.*, 7(3):359–369, 1998.

[130] Yingjie Zhang. An improved algorithm for the piecewise-smooth mumford and shah model in image segmentation. *EURASIP J. Bioinformatics and Systems Biology*, 2006(1):1-5, 2006.

[131] H. K. Zhao, T. Chan, and B. Merriman. A variational level set approach to multiphase motion. *Journal of Computational Physics*, 127(1):179–195, 1996.

[132] H. K. Zhao, S. Osher, B. Merriman, and M. Kang. Implicit and non parametric shape reconstruction from unorganized data using a variational level set method. *Computer Vision and Image Understanding*, 80(3):295–314, 2000.

[133] William P. Ziemer. *Weakly Differentiable Functions.* Springer-Verlag New York, Inc., New York, NY, USA, 1989.