A Study of Effective Variational Models and Efficient Numerical Methods for Image Registration

by

Noppadol Chumchob

UNIVERSITY OF LIVERPOOL

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

May 2010

Contents

Acknowledgements

I would like to express my deepest gratitude to my Ph.D. supervisor, Prof. Ke Chen, for his full support, guidance and patience throughout my doctoral studies at the University of Liverpool. Working with Prof. Chen has been a great pleasure and experience for me.

I further would like to thank other members of the Department of Mathematical Sciences for their advice and constructive criticism of my work: Prof. Vadim N. Biktashev, Dr. David M. Lewis, and Dr. Sebastien Guenneau.

Moreover, I would like to thank my colleagues, Dr. Carlos Brito-Loeza, Dr. Noor Badshah, and Dr. Martyn Huges for all the helpful and insightful discussions we had along this time.

Additionally, I would like to thank my family, Suchat, Uthai, Sawate, Vorachat, Chalita, and Pankamon, who patiently and lovely support me throughout my studies.

Last but not least, I am thankful to the Royal Thai Government for my financial support and the Office of Educational Affairs (the Royal Thai Embassy, London), the Office of the Higher Education Commission, and Silpakorn University for all supports throughout my doctoral studies.

Abstract

Image registration is one of the major areas of current research and applications in image processing. It is the process of finding an *optimal geometric transformation* between *corresponding images*. During the past few years, research in the field of image registration mainly falls into two categories: the design of new models for accurately registering the given images and the efficient solution of the resulting equations. The work presented in this thesis falls into both categories.

Recently, variational models have been successfully proven to be very valuable tools in a large number of image registration applications. Nonlinear systems of coupled partial differential equations (PDEs) emerge when one derives their formal Euler-Lagrange equations. As is well-known, the number of unknowns in a discretisation of the nonlinear systems can be large for high-resolution digital images. Thus, highly efficient methods become more and more important in order to perform the registration in a reasonable amount of time. Among fast iterative methods, multilevel techniques, e.g. nonlinear multigrid and multi-resolution methods, offer the potential of optimal efficiency.

This thesis presents four variational image registration models and five numerical solutions based on multilevel strategies to improve and obtain fast registration results.

First, this thesis presents a novel affine image registration model in a variational and multi-resolution framework. Several numerical tests show that the new model and the proposed numerical approach appear to be reliable and robust in i) solving the affine image registration problems and ii) providing a good initial guess for deformable image registration models.

Second, this thesis presents an efficient multigrid approach for variational image registration models based on the sum of squared differences (SSD) between images. A unified approach for designing fixed-point (FP) type smoothers is proposed and analysed by the local Fourier analysis (LFA) using Fischer–Modersitzki's diffusion and curvature image registration models [46, 47]. Numerical experiments not only show that the proposed multigrid approach is h-independent convergence, but it is also more effective than those in a large class of existing iterative methods developed by [46, 47, 48, 65, 89, 90, 131, 135, 145].

Third, this thesis presents a discontinuity-preserving image registration model based on the modified *total variation* (TV) regularisation with the so-called *potential function*. As a consequence, the new model can be simply interpreted as a half way model between the diffusive and TV regularisations for solving both smooth and non-smooth registration problems. In

order to solve the resulting Euler-Lagrange equations, several iterative methods are proposed and tested using both realistic and synthetic images. Numerical experiments show that a full approximation scheme nonlinear multigrid (FAS-NMG) method based on a new FP smoothing scheme is much faster than standard unilevel methods like semi-implicit (SI) and additive operator splitting (AOS) time marching approaches in convergence and delivering the same numerical results.

Moreover, this thesis also presents a new curvature model for solving both smooth and non-smooth registration problems. In contrast to other commonly used variational models, a theoretical result shows that the new curvature model no longer requires an additional affine linear pre-registration step. Associated with the new model is the apparent difficulty in developing a fast solution as the Euler-Lagrange equations of two coupled PDEs is highly nonlinear and of fourth order so standard unilevel methods are not appropriate. To tackle these difficulties, several FP type smoothers including the so-called *primal-dual fixed-point* (PDFP) method are proposed with a FAS-NMG framework and analysed by the LFA. As expected, the PDFP type smoother appears to be a potential smoother. Numerical tests using both synthetic and realistic images not only confirm that the proposed curvature model is more robust in registration quality for a wide range of applications than the approximate curvature models of [47, 48, 79, 78, 73, 75, 74], but also that the FAS-NMG approach based on the proposed PDFP type smoother is fast and accurate in delivering visually-pleasing registration results.

Finally, this thesis presents an improved monomodal image registration model combining a non-parametric intensity and geometric transformation, as an alternative model to using mutual information for a typical case of multimodal images where the given images have the similar features, but different intensity variations. It is modelled by modifying the sum of squared differences and applying the new curvature model to constrain both intensity and geometric transformations. In order to solve the resulting Euler-Lagrange equations, this work extends the PDFP type smoother and uses as a recommended smoother for a FAS-NMG approach. Compared with the variational model introduced by [106], numerical results show that the new registration model and the FAS-NMG approach based on the PDFP type smoother are reliable to provide satisfactory registration results for practical applications.

Overall this thesis is concerned with effective variational model and efficient numerical methods for image registration.

Publications and Presentations

Publications

- N. Chumchob and K. Chen, *A Robust Affine Image Registration Method*, International Journal of Numerical Analysis and Modeling, 6(2): 311-334, 2009.
- N. Chumchob and K. Chen, *A Robust Multigrid Approach for Variational Image Registration Models*, Submitted to Journal of Computational and Applied Mathematics, 2010.
- N. Chumchob and K. Chen, *A Variational Approach for Discontinuity-Preserving Image Registration*, Submitted to East-West Journal of Mathematics, 2010.
- N. Chumchob, K. Chen, and C. Brito-Loeza, *A Fourth Order Variational Image Registration Model and Its Fast Multigrid Algorithm*, Submitted to Multiscale Modeling and Simulation, 2010.
- N. Chumchob and K. Chen, *An Improved Monomodal Image Registration Model and Its Fast Algorithm*, In preparation.

Presentations

- **N. Chumchob** and K. Chen, *A Variational Approach for Discontinuity-Preserving Image Registration*, International Conference in Mathematics and Applications, Bangkok, Thailand, December 17th-19th, 2009.
- **N. Chumchob**, K. Chen, and C. Brito-Loeza, *A Fourth Order Nonlinear PDE-Based Image Registration Model and Its Fast Algorithm*, SIAM Conference on Imaging Science (IS10), Chicago, Illinois (USA), April 12th-14th, 2010.

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction to image registration

One of the major problems in current image processing is *image registration*, sometimes also called *image fusion*, *image matching* or *image warping*. It is the process of finding an *optimal geometric transformation* between *corresponding* images. It can also be viewed as the process of overlaying two or more images of the same or similar scene taken at different times, from different perspectives, and/or by different imaging machineries. Therefore, this procedure is required whenever a series of corresponding images needs to be compared or integrated.

Figure 1.1: Two squares, LEFT: reference image, RIGHT: template image.

More precisely, the image registration problem can be phrased generally in only a few words: given a so-called *reference* and a so-called *template* image, find a suitable *transformation* such that the transformed template image becomes *similar* to the reference. Unfortunately, the problem is easy to state, but it is hard to solve. The main reason is that the image registration problem is known to be a highly ill-posed one in the sense of Hadamard as will be explained in §2.4 and §3.2. A more subtle point is illustrated in Figure 1.1[1], showing a white rectangle on the black background. For simplicity, we only allow rigid-body transformations, the special type of affine linear transformations consists of translation and rotation. We immediately find two different solutions: a pure translation and a translation followed by a rotation of 360 degrees around the center of the image. These solutions are equivalent. Without additional knowledge it is not possible to decide which one to use. Therefore, a typical treatment has to be taken, as

[1] Adapted from [104, p. 3]

discussed later in §3.2.

Image registration can be classified broadly into two main *physical* categories: rigid and non-rigid registration, or *mathematical* categories: linear and nonlinear registration, or *complexity* categories: parametric and non-parametric registration. In several applications, rigid registration approaches, which involve a rigid-body transformation (with only 3 unknown parameters for 2 dimensional problems), cannot always provide a satisfactory result, particularly in medical applications (e.g. deformable or soft tissue images), while on the other hand non-rigid or deformable registration approaches may not be quick enough for ready use (e.g. variational registration models reviewed by [104] may include nonlinear transformations (non-parametric) whose number of unknowns for a discrete image is proportional to the number of image pixels!). Therefore, it is still a challenge to design a robust registration model and its numerical solutions suitable for real-life applications. For an overview on image registration approaches, we refer to [16, 49, 68, 100, 104, 124] and references therein.

1.2 Applications of image registration

There are several applications that require a registration step ranging from art, astronomy, astro-physics, biology, chemistry, criminology, genetic, physics, and other areas involving imaging techniques. More specific applications are, for example, remote sensing (registration of satellite images taken over a region during different seasons or years can be used to detect environment change over time), security (comparing current images with a data base), robotic (tracking of objects), and in particular medicine, where computational anatomy, computer-aided diagnosis, fusion of different modalities, intervention and treatment planning, monitoring of diseases, motion correction, radiation therapy or treatment verification demand registration. Since imaging techniques, like computer tomography (CT), diffusion tensor imaging (DTI), magnetic resonance imaging (MRI), positron emission tomography (PET), single-photo emission computer tomography (SPECT) or ultrasound (US) under a fascinating or ongoing improvement in the last decade, a tremendous increase in the utilisation of the various modalities in medicine takes place; see also in [16, 41, 42, 49, 56, 68, 82, 100, 124, 116].

1.3 Image registration studies - Chapters of this thesis

Research in the field of image registration mainly falls into two categories: the design of new models for accurately registering the given images and the efficient solution of the resulting equations. The work presented in this thesis falls into both categories and is organized as follows:

- **Chapter 2** provides various mathematical tools which will be used throughout the rest of the thesis. It includes:
  - Useful preliminary definitions, theorems, and examples of normed linear spaces and functions of bounded variation.

- An introduction into calculus of variations.
- A brief discussion of ill-posed inverse problems and regularisation.
- A discussion of the discretisation of partial differential equations (PDEs) on regular domains using finite difference methods.
- A review of basic iterative methods for solving linear and nonlinear systems.
- An introduction into multigrid methods as iterative solvers for discrete elliptic PDEs including algorithms for linear and nonlinear multigrid methods.

- **Chapter 3** details a state-of-the-art registration framework. It includes:
  - The general variational formulation of image registration.
  - A brief review of similarity measures, including the sum of squared differences (SSD) and the mutual information (MI).
  - A brief review and discussion of deformation models (regularisations) commonly used in deformable image registration, including the elastic model [8, 15, 104], the diffusion model [33, 46, 89, 91, 104, 131], the Fischer and Modersitzki's cuvature model [47, 48, 49, 89, 91, 104], the Henn and Witsch's curvature model [79, 78, 73, 75, 74], and the total variation (TV) model [51, 53, 142].
  - A discussion of general solution schemes, including the optimise-discretise and discretise-optimise approaches.
  - A brief survey of multigrid methods for deformable image registration.

- **Chapter 4** presents a robust affine image registration (RAR) method in a variational and multi-resolution framework. It includes:
  - Details of affine image registration, solution methods by the Gauss-Newton (GN) and Levenberg-Marquardt (LM) approaches, and some registration results using the GN and LM methods.
  - Details of four initialisation techniques to improve the GN and LM methods.
  - Details of the RAR method in the variational and multi-resolution framework.
  - Tests showing the robustness of the RAR method.

- **Chapter 5** presents a robust multigrid approach for variational image registration models. It includes:
  - A discussion of the nonlinear fitting term that restricts the class of numerical methods for variational image registration models.
  - Details of the diffusion model and its numerical methods, including a discretisation and a brief review of previous works on non-multigrid and multigrid methods.

- Details of the proposed nonlinear multigrid method for the diffusion model, including the new and robust fixed-point (FP) smoother and its smoothing analysis by the LFA (local Fourier analysis).
- Details of the curvature model and its efficient FP method that we have been used as a potential smoother in the nonlinear multigrid framework.
- Tests showing the effectiveness of the proposed nonlinear multigrid methods with the new FP smoothers for the diffusion and curvature models, including several comparisons with other numerical approaches commonly used in the literatures.

- **Chapter 6** presents a discontinuity-preserving image registration model and its fast solution. It include:
  - A discussion of the commonly used regularisation methods that provide either smooth or non-smooth deformation fields.
  - Details of the proposed variational model for preserving discontinuities of deformations fields.
  - Details of a finite difference discretisation and four numerical solutions of the resulting equations.
  - Details of the proposed nonlinear multigrid method.
  - Tests showing the effectiveness of the proposed model and multigrid method, including several comparisons with other registration models and numerical solutions.

- **Chapter 7** presents a fourth-order variational image registration model and its fast multigrid method. It includes:
  - A discussion of the commonly used PDE-based image registration models that provide either smooth or non-smooth deformation fields.
  - Details and discussions of a new PDE model resulting from the proposed curvature regularisation.
  - A discussion of five numerical solutions of the resulting PDE system.
  - Details of the proposed nonlinear multigrid method, including the LFA of the new smoother and the nonlinear multigrid algorithm.
  - Tests showing the robustness of the new PDE model and the proposed nonlinear multigrid method.

- **Chapter 8** presents an improved monomodal image registration model and its fast solution. It includes:
  - A review and discussion of image registration models combining an intensity and geometric transformation, as an alternative way to using mutual information for a typical case of multimodal images having the similar features, but different intensity variations.

- Details of the proposed variational model, including the Euler-Lagrange system and its primal-dual formulation.
- Details of the numerical solution for the primal-dual formulation, including a potential FP method and a nonlinear multigrid algorithm.
- Tests showing the robustness of the new variational model and the proposed nonlinear multigrid method.

- **Chapter 9** summarises our work and covers possible future research directions.

All experiments presented in the thesis were run in MATLAB R2008a on a Dell Precision T7400 with quad-core Intel Xeon processors and 4 Gigabytes of RAM.

Chapter 2

Mathematical Preliminaries

This chapter introduces various materials which will be used throughout the rest of the thesis.

2.1 Normed linear spaces

Definition 2.1.1 (*Linear vector space*). *A linear vector space over a field F (usually of real or complex numbers) is a set V together with two binary operations, operations that combine two entities to yield a third, called vector addition and scalar multiplication such that, the following conditions hold:*

1. *Closure of vector addition: If $u, v \in V$, then $u + v \in V$.*
2. *Commutativity of addition: If $u, v \in V$, then $u + v = v + u$.*
3. *Associativity of addition: If $u, v, w \in V$, then $(u + v) + w = u + (v + w)$.*
4. *Identity element of addition: There exists an element $0 \in V$, called the zero vector, such that $v + 0 = v$ for all $v \in V$.*
5. *Existence of additive inverse: For each $u \in V$, there exists $-u \in V$ such that $u + (-u) = 0$.*
6. *Closure of scalar multiplication: If $\lambda \in F$ and $u \in V$, then $\lambda u \in V$.*
7. *Associativity of scalar multiplication: If $u \in V$ and $\lambda, \theta \in F$, then $\lambda(\theta u) = (\lambda\theta)u$.*
8. *Scalar multiplication is distribute: If $u, v \in V$ and $\lambda, \theta \in F$, then $(\lambda + \theta)u = \lambda u + \theta u$ and $\lambda(u + v) = \lambda u + \lambda v$.*
9. *Identity element of scalar multiplication: There exists an element $1 \in V$, called the multiplicative identity, such that $1v = v$ for all $v \in V$.*

A subset of a linear vector space V which is also a linear vector space over the same field and under the same operators of addition and scalar multiplication is called a *linear subspace* of V.

Example 2.1.1 *Examples of linear vector spaces are*

- *The spaces $\mathbb{R}^d$ and $\mathbb{C}^d$ for all $d \in \mathbb{N}$.*
- *The space $\mathcal{C}^k(\Omega, \mathbb{R}^d)$ of all functions on the domain $\Omega \subset \mathbb{R}^d$ whose partial derivatives of order up to k are continuous.*

Definition 2.1.2 (*Norm*). *A norm on a linear vector space V is a real-valued function $\|\cdot\|$ defined on V such that*

1. *$\|u\| > 0$ if $u \neq 0$.*
2. *$\|\lambda u\| = |\lambda| \, \|u\|$ for all scalars λ and vectors u.*
3. *$\|u + v\| \leq \|u\| + \|v\|$ for all $u, v \in V$.*

A semi-norm is defined similarly to above except that axiom 1 is replaced by $\|u\| \geq 0$, and therefore it is possible for a semi-norm to be equal to zero for some $u \neq 0$.

Definition 2.1.3 (*Normed linear space*). *A normed linear space is a linear vector space V equipped with a norm $\|\cdot\|$.*

Example 2.1.2 (*p-norm*). *Consider $\mathbf{x} \in \mathbb{R}^d$, then for any real number $p \geq 1$ the p-norm of $\mathbf{x}$ is defined by*

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

where for $p = 2$ we recover the Euclidean norm defined by

$$\|\mathbf{x}\|_{\mathbb{R}^d} = \sqrt{\mathbf{x} \cdot \mathbf{x}} = \sqrt{\sum_{i=1}^{n} x_i^2}.$$

Note that the p-norm can be extended to vectors having an infinite number of components, yielding the l^p-space defined as the set of all infinite sequences of real or complex numbers with finite p-norm.

Example 2.1.3 (*L^p-norm*). *Consider a function f defined on a domain Ω and $1 \leq p \leq \infty$. Then*

$$\|f\|_{L^p} = \left(\int_\Omega |f(\mathbf{x})|^p d\mathbf{x} \right)^{1/p}$$

defines the L^p-norm of f on Ω. Note that this is a generalisation of the previous example since f is now allowed to have not only countably-infinitely many components but arbitrarily many components. The spacial case when $p = \infty$ is defined as

$$\|f\|_{L^\infty} = \sup_{\mathbf{x}} |f(\mathbf{x})|.$$

Definition 2.1.4 (*Inner product*). *An inner product on a linear vector space V is a function $\langle \cdot, \cdot \rangle_V$ defined on $V \times V$ which satisfies:*

1. *$\langle u, v \rangle_V = \overline{\langle v, u \rangle}_V$ for all $u, v \in V$*

2. $\langle \lambda u, v \rangle_V = \lambda \langle u, v \rangle_V$.

3. $\langle u + v, w \rangle_V = \langle u, w \rangle_V + \langle v, w \rangle_V$.

4. $\langle u, u \rangle_V > 0$ *when* $u \neq 0$.

Example 2.1.4 *The classical example of an inner product is the function* $\langle \cdot, \cdot \rangle_{\mathbb{R}^d}$ *defined on* $\mathbb{R}^d \times \mathbb{R}^d$ *by* $\langle \mathbf{x}, \mathbf{y} \rangle_{\mathbb{R}^d} = \mathbf{y}^\top \mathbf{x} = \sum_{i=1}^{n} x_i y_i$ *for all* $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$.

We note that any inner product on a linear vector space V induces a norm defined by $\|u\|_V = \langle u, u \rangle_V^{1/2}$. This also means that an inner product space, a linear vector space together with an inner product defined on it, is a special type of normed spaces.

Definition 2.1.5 (*Cauchy sequence*). *A sequence* $\{u_k\}_{k=1}^{\infty}$ *in a normed linear space is said to be a Cauchy sequence if for all* $\epsilon > 0$ *there exists a* $K_0 \in \mathbb{N}$ *such that any* $k, l \geq K_0$ *implies that* $\|u_k - u_l\| < \epsilon$.

Definition 2.1.6 (*Banach space*). *A normed linear space* L *is said to be complete if every Cauchy sequence in* L *converges to an element in* L. *A complete normed linear space is called Banach space.*

Similarly, a complete inner product space is known as *Hibert space*. Two relevant examples of Hilbert spaces are the space $\mathbb{R}^d$ together with the Euclidean inner product and the space $L^2(\Omega)$ together with the inner product defined by $\langle f, g \rangle_{L^2(\Omega)} = \int_\Omega f(\mathbf{x}) g(\mathbf{x}) d\mathbf{x}$.

Definition 2.1.7 (*Linear operator*). *An operator* $A : V \to W$, *where* V *and* W *are vector spaces, is linear if* $A(av_1 + bv_2) = aAv_1 + bAv_2$ *for all* $v_1, v_2 \in V$ *and all scalars* a, b.

Example 2.1.5 *A linear operator mapping* $\mathbb{R}^n$ *to* $\mathbb{R}^m$ *is defined by a matrix* $\mathbf{A}$ *of size* $m \times n$, *then given* $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{y} = \mathbf{A}\mathbf{x} \in \mathbb{R}^m$.

Definition 2.1.8 (*Convex set*). *A set* S *is convex if for all* $u, v \in S$

$$\lambda u + (1 - \lambda) v \in S$$

for all $\lambda \in [0, 1]$.

Definition 2.1.9 (*Convex function*). *A function* f *defined on a convex set is convex if for all* $u, v \in S$

$$f(\lambda u + (1 - \lambda) v) \leq \lambda f(u) + (1 - \lambda) f(v)$$

for all $\lambda \in [0, 1]$. *It is called strictly convex provided that the strict inequality holds for* $x \neq y$ *and* $\lambda \in (0, 1)$.

Example 2.1.6 *The total variation (TV) of* $u : \Omega \subset \mathbb{R}^2 \to \mathbb{R}$ *denoted by* $TV(u)$ *and defined as follows:*

$$TV(u) = \int_\Omega |\nabla u(\mathbf{x})| d\mathbf{x}$$

is convex; see more details for the TV of u *in* §2.3.

2.2 Introduction into calculus of variations

In this section we address a class of minimisation problems where we search for an appropriate function rather than a value of some variable, that makes a given quantity (usually an energy or action integral) stationary. Because a function is varied, these problems are called *variational* and solved by the so-called *calculus of variations*. The calculus of variations involves problems in which the quantity to be minimised (or maximised) appears as a certain definite integral of an unknown function and/or its derivatives.

Consider the general minimisation problem

$$\min_{u} \mathcal{J}(u), \tag{2.1}$$

where $\mathcal{J} : \mathcal{U} \to \mathbb{R}$ be a general functional[1] and $\mathcal{U}$ denotes a solution space consisting of admissible functions minimising $\mathcal{J}$ (for example, $u \in \mathcal{C}^2(\Omega, \mathbb{R}^d)$ with $u = u_0$ on $\partial\Omega$). We denote by $\mathcal{V}$ a *test space* consisting of all functions which can be written as the difference between any two admissible functions,

$$\mathcal{V} = \{ v | \, v = u - \widehat{u} \text{ and } u, \widehat{u} \in \mathcal{U} \}. \tag{2.2}$$

We start first by describing a particular subset of $\mathcal{U}$.

Definition 2.2.1 (*Neighbourhood*). *Given a solution space $\mathcal{U}$, a function $\widehat{u} \in \mathcal{U}$, and $\varepsilon > 0$, then $\mathcal{B}_\varepsilon(\widehat{u})$ denotes the neighbourhood of $\widehat{u}$ as*

$$\mathcal{B}_\varepsilon(\widehat{u}) = \{ u \in \mathcal{U} | \, \|u - \widehat{u}\| < \varepsilon \}.$$

With the general functional given by (2.1), a local minimiser can be defined as follows.

Definition 2.2.2 (*Local minimiser*). *Given a solution space $\mathcal{U}$ and a functional $\mathcal{J} : \mathcal{U} \to \mathbb{R}$, $\widehat{u} \in \mathcal{U}$ is said to be a local minimiser of $\mathcal{J}$ if for every $\varepsilon > 0$ there exists a $\delta > 0$ such that $\mathcal{J}(\widehat{u}) \leq \mathcal{J}(u)$ for all $u \in \mathcal{B}_\varepsilon(\widehat{u})$.*

To define the necessary condition for a local minimiser of $\mathcal{J}$, the existence of a directional derivative is required.

Definition 2.2.3 (*Gâteaux derivative*). *Given a solution space $\mathcal{U}$, a test space $\mathcal{V}$, and a functional $\mathcal{J} : \mathcal{U} \to \mathbb{R}$, $\mathcal{J}$ is Gâteaux-differentiable for $u \in \mathcal{U}$ in the direction of $v \in \mathcal{V}$ if*

1. *there exists a number $\widehat{\epsilon} > 0$ such that $u_\epsilon = u + \epsilon v \in \mathcal{U}$ for all $|\epsilon| \leq \widehat{\epsilon}$, and*
2. *the function $J(\epsilon) = \mathcal{J}(u_\epsilon)$ is differentiable at $\epsilon = 0$.*

The first order Gâteaux derivative (or directional derivative or first variation) of $\mathcal{J}$ for u in the direction of v is defined by

$$\delta\mathcal{J}(u; v) = J'(0) = \left. \frac{d\mathcal{J}(u + \epsilon v)}{d\epsilon} \right|_{\epsilon=0} = \lim_{\epsilon \to 0} \frac{\mathcal{J}(u + \epsilon v) - \mathcal{J}(u)}{\epsilon}.$$

[1] A function which depends on one or more functions rather than on discrete variable is referred to as a *functional*.

Now, a stationary point can be defined as follows.

Definition 2.2.4 (*Stationary point*). *Let a solution space $\mathcal{U}$, a test space $\mathcal{V}$, and a functional $\mathcal{J} : \mathcal{U} \to \mathbb{R}$ be given. Suppose that for some $\widehat{u} \in \mathcal{U}$, $\mathcal{J}$ is Gâteaux-differentiable for all test function $v \in \mathcal{V}$. Then $\widehat{u}$ is said to be a stationary point of $\mathcal{J}$ if $\delta\mathcal{J}(\widehat{u}; v) = 0$ for all $v \in \mathcal{V}$.*

We now give a necessary condition for a minimiser which can be formulated by linking a stationary point to a minimiser.

Theorem 2.2.1 (*Necessary condition for a local minimiser*). *Let a solution space $\mathcal{U}$ with an admissible function $\widehat{u} \in \mathcal{U}$, a functional $\mathcal{J} : \mathcal{U} \to \mathbb{R}$, and a test space $\mathcal{V}$ be given. Assume that $\mathcal{J}$ is Gâteaux-differentiable for $\widehat{u}$ and all test function $v \in \mathcal{V}$. Then:*

If $\widehat{u}$ is a local minimiser of $\mathcal{J}$, then $\widehat{u}$ is a stationary point of $\mathcal{J}$.

Proof. The proof of this theorem can be found in [5, 36, 88]. ■

With this theorem we can investigate the condition for a stationary point of some functional $\mathcal{J}$ in more detail. Here we specify and consider the general functional $\mathcal{J}$ defined by

$$\mathcal{J}(u) = \int_{\Omega} F[\mathbf{x}, u(\mathbf{x}), \nabla u(\mathbf{x})] d\mathbf{x}, \tag{2.3}$$

where $\Omega \subset \mathbb{R}^d$, $d \geq 1$, is a bounded open set and F is a functional depending on $\mathbf{x} = (x_1, \ldots, x_d)^\top$, $u : \mathbb{R}^d \to \mathbb{R}$, and $\nabla u(\mathbf{x}) = (\partial u/\partial x_1, \ldots, \partial u/\partial x_d)^\top$. Assume that $\mathcal{J}$ is Gâteaux-differentiable in all directions of the respective test space. Thus F is assumed to have continuous partial derivatives with respect to its arguments.

We introduce the following notation before giving the condition for a stationary point of $\mathcal{J}$. We denote

$$\nabla_u F = \partial F/\partial u = F_u \tag{2.4}$$

to be the gradient of F respect to u to distinguish the usual gradient denoted by $\nabla F = (\partial F/\partial x_1, \ldots, \partial F/\partial x_d)^\top$. In similar way, the gradient of F with respect to ∇u is given by

$$\nabla_{\nabla u} F = (\partial F/\partial u_{x_1}, \ldots, \partial F/\partial u_{x_d})^\top \in \mathbb{R}^d. \tag{2.5}$$

For this moment we will restrict the solution space $\mathcal{U}$ by prescribing a specific boundary condition, i.e.

$$\widetilde{\mathcal{U}} = \{ u \in \mathcal{U} | \, u = c \text{ on } \partial\Omega \} \tag{2.6}$$

and then the corresponding test space is given by

$$\widetilde{\mathcal{V}} = \{ v \in \mathcal{V} | \, v = 0 \text{ on } \partial\Omega \}. \tag{2.7}$$

However, the following result can not only be extended to the general spaces $\mathcal{U}$ and $\mathcal{V}$, but also to the vectorial case where $\boldsymbol{u} = (u_1, u_2, \ldots, u_d)^\top : \mathbb{R}^d \to \mathbb{R}^d$.

Lemma 2.2.1 (*Stationary of $\mathcal{J}$*). *A function $u \in \widetilde{\mathcal{U}}$ is a stationary point of the general functional $\mathcal{J}$ (2.3) if*

$$\int_\Omega \langle \nabla_u F - \nabla \cdot \nabla_{\nabla u} F, v \rangle_{\mathbb{R}^d} \, d\mathbf{x} = 0 \tag{2.8}$$

holds for all test function $v \in \widetilde{\mathcal{V}}$.

Proof. Let $\epsilon \in \mathbb{R}$. Setting the Gâteaux derivative of $\mathcal{J}$ for u in the direction of v to zero leads to

$$\begin{aligned}
0 = \delta \mathcal{J}(u; v) &= \left. \frac{d\mathcal{J}(u + \epsilon v)}{d\epsilon} \right|_{\epsilon=0} \\
&\overset{*}{=} \int_\Omega \left. \frac{dF[\mathbf{x}, u(\mathbf{x}) + \epsilon v(\mathbf{x}), \nabla u(\mathbf{x}) + \epsilon \nabla v(\mathbf{x})]}{d\epsilon} \right|_{\epsilon=0} d\mathbf{x} \\
&= \int_\Omega \left. \frac{\partial F}{\partial (u + \epsilon v)} \frac{\partial (u + \epsilon v)}{\partial \epsilon} \right|_{\epsilon=0} \\
&\quad + \sum_{m=1}^{d} \left(\frac{\partial F}{\partial (u_{x_m} + \epsilon v_{x_m})} \frac{\partial (u_{x_m} + \epsilon v_{x_m})}{\partial \epsilon} \right) \Bigg|_{\epsilon=0} d\mathbf{x} \\
&= \int_\Omega \frac{\partial F}{\partial u} v + \sum_{m=1}^{d} \frac{\partial F}{\partial u_{x_m}} v_{x_m} d\mathbf{x} \qquad \text{for all } v \in \widetilde{\mathcal{V}},
\end{aligned}$$

where we used in $(*)$ an interchange of differentiation and integration followed by application of the chain rule. By using the Gauss (divergence) theorem we get

$$\begin{aligned}
0 &= \int_\Omega \frac{\partial F}{\partial u} v d\mathbf{x} - \int_\Omega \sum_{m=1}^{d} \frac{\partial}{\partial_{x_m}} \frac{\partial F}{\partial u_{x_m}} v d\mathbf{x} + \int_{\partial\Omega} \langle \nabla_{\nabla u} F, \boldsymbol{n} \rangle_{\mathbb{R}^d} \, v d\mathbf{x} \\
&= \int_\Omega \langle \nabla_u F - \nabla \cdot \nabla_{\nabla u} F, v \rangle_{\mathbb{R}^d} \, d\mathbf{x} + \int_{\partial\Omega} \langle \nabla_{\nabla u} F, \boldsymbol{n} \rangle_{\mathbb{R}^d} \, v d\mathbf{x}
\end{aligned} \tag{2.9}$$

holding for all test functions $v \in \widetilde{\mathcal{V}}$. Since every test function fulfills $v = 0$ on $\partial\Omega$, the boundary integral vanishes and the proof is completed. ■

It is clear that (2.8) holds for an arbitrary test function only if $\nabla_u F - \nabla \cdot \nabla_{\nabla u}$ vanishes. This assertion is included in the well-known theorem; see [5, 36, 88] for example. Therefore, $u \in \widetilde{\mathcal{U}}$ is a stationary point of the Gâteaux-differentiable functional $\mathcal{J}$ (2.3) if

$$\nabla_u F - \nabla \cdot \nabla_{\nabla u} F = 0 \text{ on } \Omega. \tag{2.10}$$

By applying Theorem 2.2.1 (2.10) is then a necessary condition for a (local) minimiser of (2.1). Typically $d > 1$ and (2.10) leads to a (partial) differential equation, known as the *Euler-Lagrange equation*. Together with boundary conditions, e.g. described by $\widetilde{\mathcal{U}}$, we are faced with a boundary value problem with the minimisation problem in (2.1) called its *variational formulation* [5, 36, 88]. If the boundary conditions are imposed explicitly on a solution space $\mathcal{U}$ as in Lemma 2.2.1, they are called *essential condition*. If, in contrast, boundary conditions are not given explicitly in the definition of $\mathcal{U}$, we are dealing with *natural* conditions. These conditions depend on the general functional $\mathcal{J}$ or, to be exact, on its integrand F. For an illustration we recall (2.9). In its context we discussed the circumstances under which

$$\int_{\partial\Omega} \langle \nabla_{\nabla u} F, \boldsymbol{n} \rangle_{\mathbb{R}^d} \, v d\mathbf{x} = 0 \tag{2.11}$$

holds when using the restricted spaces $\widetilde{\mathcal{U}}$ and $\widetilde{\mathcal{V}}$. However, the same equality can be achieved when using the general spaces $\mathcal{U}$ and $\mathcal{V}$, for instance with a boundary condition on a part of the boundary only or even without any boundary condition.

In summary every solution $u^* \in \mathcal{U}$ of the general minimisation problem (2.1) with a Gâteaux-differentiable functional $\mathcal{J}$ as given by (2.3) is a solution of the boundary value problem consisting of the Euler-Lagrange equation

$$\nabla_u F - \nabla \cdot \nabla_{\nabla u} F = 0 \text{ on } \Omega$$

subject to boundary conditions which can be either the essential type (when incorporated in the definition of the solution space $\mathcal{U}$) or natural type

$$\langle \nabla_{\nabla u} F, \boldsymbol{n} \rangle_{\mathbb{R}^d} = 0 \text{ on } \partial\Omega.$$

Here, $\boldsymbol{n}=(n_1, \ldots, n_d)^\top$ denotes the outer normal unit vector of $\partial\Omega$.

Example 2.2.1 *Let $d = 2$, $\Omega = [0,1]^2$, $F = |\nabla u|^2$ where $u = u(\mathbf{x})$. The variational formulation of*

$$\min_u \int_\Omega |\nabla u|^2 \, d\mathbf{x}$$

is equivalent to the boundary value problem given by

$$\begin{aligned} -\Delta u &= 0 \text{ on } \Omega \\ \frac{\partial u}{\partial \boldsymbol{n}} &= 0 \text{ on } \partial\Omega. \end{aligned}$$

Example 2.2.2 *Let $d = 2$, $\Omega = [0,1]^2$, $F = |\nabla u|$ where $u = u(\mathbf{x})$. The variational formulation of*

$$\min_u \int_\Omega |\nabla u| \, d\mathbf{x}$$

is equivalent to the boundary value problem given by

$$\begin{aligned} -\nabla \cdot \frac{\nabla u}{|\nabla u|} &= 0 \text{ on } \Omega \\ \frac{\partial u}{\partial \boldsymbol{n}} &= 0 \text{ on } \partial\Omega. \end{aligned}$$

2.3 Functions of bounded variation

Let Ω be a bounded open subset of $\mathbb{R}^d$ and let $u \in L^1(\Omega)$. Define the total variation of u as

$$\begin{aligned} \int_\Omega |Du| = \sup\{ \int_\Omega u \nabla \cdot \varphi d\mathbf{x} |\ \boldsymbol{\varphi} = (\varphi_1, \varphi_2, \ldots, \varphi_d) \in \mathcal{C}_0^1(\Omega, \mathbb{R}^d)^d \\ \text{and } ||\varphi_i||_{L^\infty} \leq 1 \text{ for } i = 1, \ldots d \}, \end{aligned} \tag{2.12}$$

where $\nabla \cdot \boldsymbol{\varphi} = \sum_{i=1}^{d} \frac{\partial \varphi_i}{\partial x_i}(\mathbf{x})$, $d\mathbf{x}$ is the Lebesgue measure[2] and $\mathcal{C}_0^1(\Omega, \mathbb{R}^d)$ is the space of continuously differentiable functions with compact support in Ω.

[2]In Euclidean space, the standard way to assign a *measure* (length, area or volume) to a given subset is through the Lebesgue measure. Hence, sets with finite Lebesgue measure and called Lebesgue measurable. In real analysis, this measure is used to define Lebesgue integration.

As described in [55] for a particular and interesting case of $u \in \mathcal{C}^1(\Omega, \mathbb{R}^d)$, integration by parts gives

$$\int_\Omega u \nabla \cdot \boldsymbol{\varphi} d\mathbf{x} = -\int_\Omega \sum_{i=1}^{d} \frac{\partial u}{\partial x_i} \varphi_i d\mathbf{x} \tag{2.13}$$

for every $\boldsymbol{\varphi} \in \mathcal{C}_0^1(\Omega, \mathbb{R}^d)^d$ and

$$\int_\Omega |Du| = \int_\Omega |\nabla u| \, d\mathbf{x}. \tag{2.14}$$

A function $u \in L^1(\Omega)$ is said to have bounded variation in Ω if $\int_\Omega |Du| < \infty$. We define $BV(\Omega)$ as the space of all functions in $L^1(\Omega)$ with bounded variation.

Example 2.3.1 *The following functions $f1$, $f2$ and $f3$ defined by*

$$f1(x) = \sin x, \tag{2.15}$$

$$f2(x) = \begin{cases} 1/4, \ for \ x \in [0, \pi/8) \\ 1/2, \ for \ x \in [\pi/8, \pi/4) \\ 3/4, \ for \ x \in [\pi/4, 3\pi/8) \\ 1, \ for \ x \in [3\pi/8, \pi/2] \end{cases}, \tag{2.16}$$

$$f3(x) = \frac{2x}{\pi}, \tag{2.17}$$

belong to $BV(\Omega)$ with $\Omega = [0, \pi/2]$ and have the same total variation equal to one. The function $f4$ defined by

$$f4(x) = \begin{cases} 0, \ for \ x = 0 \\ \sin 1/x, \ for \ x \in (0, a) \ with \ a > 0 \end{cases} \tag{2.18}$$

has infinite total variation and does not belong to $BV(\Omega)$ with $\Omega = [0, a]$ for all $a > 0$.

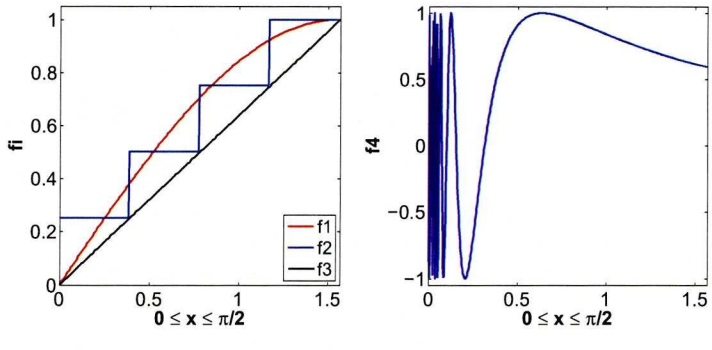


Figure 2.1: Left: three bounded variation functions with the same total variation equal to one. Right: a function with infinite total variation.

We shall conclude this subsection by giving the coarea formula.

Theorem 2.3.1 (*Coarea formula*). *Let $\Omega \subset \mathbb{R}^d$ be an open set and let $u \in BV(\Omega)$. Let $L_\lambda = \{\mathbf{x} \in \Omega |\ u(\mathbf{x}) < \lambda\}$ be the level domain. Then*

$$\int_\Omega |Du| = \int_{-\infty}^{\infty} Per(L_\lambda, \Omega) d\lambda,$$

where $Per(L_\lambda, \Omega) = \int_\Omega |D_{\chi L_\lambda}|$ is the perimeter of L_λ in Ω and χL_λ is a characteristic (or indicator) function of L_λ.

Proof. The proof could be found in [55]. ■

2.4 Ill-posed inverse problems and regularisation

Ill-posed inverse problems are formulated and solved on a daily basis in several areas such as astrophysics, geophysics, and in particular image processing. A simple way to visualize an inverse problem is to imagine that we are given a black box and we wish to find out its contents. We are not allowed to open it but we are allowed to carry out any experiments for output data to find the unknown input creating this data. In inverse problems we call the unknown input as *the solution* or *model* and the results of an experiment as *the data*. The experiment itself is referred as *the forward modelling*. Usually an experiment cannot guarantee to determine a unique solution, i.e. there could be more than one solution which would produce the same data. In order to select the most reasonable solution, we need to impose a constraint which is known mathematically as *regularisation*. Regularisation produces a solution satisfying some specific criteria using *priori* information and penalises unwanted solutions.

In mathematics we have a classical definition of an ill-posed problem. According to the famous French mathematician Hadamard (1902) a problem is *ill-posed* if one of the following conditions does not hold

- the solution exists
- the solution is unique
- the solution depends continuously on the data (i.e. a small change in the data does not lead to a large change in the solution)

and is *well-posed* if all conditions are satisfied. However, Hadamard did not deal with any numerical solutions of ill-posed problems as he believed that the ill-posedness arose from an incorrect physical representation of the problem. In 1963 the Russian mathematician Andrei N. Tikhonov introduced the foundations of the theory of ill-posed problem solutions and he developed the concept of regularisation which was based on an approximation of an ill-posed problem by a number of well-posed problems.

Example 2.4.1 *Let $\mathcal{H}$ be a Hilbert space, let $m(t) \in \mathcal{H}$ be the model and let $\mathbf{d} = (d_1, \ldots, d_N)^\top \in \mathbb{R}^N$ be a vector of the measured data. Suppose that the relation between m and $\mathbf{d}$ is given by*

the Fredholm integral equation of the first kind as follows:

$$d_i = \int_D K(s_i;t)m(t)dt + \epsilon_i \tag{2.19}$$

where $K(s;t)$ is a smooth kernel (i.e. the kernel does not posses singularities), ϵ_i is the measurement noise assumed to be Gaussian with mean 0 and standard deviation σ, and D is the domain of integration. Our goal is to find the model m from the given noisy data. By some quadrature rule:

$$\int_D K(s_j;t)m(t)dt \approx \sum_{i=1}^{M} w_i K(s_j,t_i)m(t_i)\Delta t_i \tag{2.20}$$

it leads to the discrete system

$$\mathbf{d} = \mathbf{A}\mathbf{u} + \epsilon \tag{2.21}$$

where $\mathbf{A}_{ji} = w_i K(s_j,t_i)\Delta t_i$ and $\mathbf{u} = m(t_i) \in \mathbb{R}^M$. We can see that the matrix $\mathbf{A} : \mathbb{R}^M \to \mathbb{R}^N$ is typically ill-conditioned since the data contain noise. Therefore a regularisation method is needed for the solution of the problem.

Let us try to select from all possible solutions the one which is the simplest in some sense, for example, it has the smallest Euclidean norm:

$$\mathcal{R}(\mathbf{u}) = \|\mathbf{u}\|^2 = \mathbf{u}^\top \mathbf{u}. \tag{2.22}$$

Thus we have the following problem: find $\mathbf{u}$ that minimises $\mathcal{R}(\mathbf{u})$ subject to the constraint

$$\mathcal{D}(\mathbf{u}) = \|\mathbf{d} - \mathbf{A}\mathbf{u}\|^2 = \|\epsilon\|^2 = \sum_{i=1}^{N} \epsilon_i^2 = \sigma^2 \sum_{i=1}^{N} \left(\frac{\epsilon_i}{\sigma}\right)^2 = \sigma^2 N = T. \tag{2.23}$$

We then can transform this problem into an optimisation problem:

$$\min_{\mathbf{u}} \left\{ \mathcal{J}_\alpha(\mathbf{u}) = \mathcal{R}(\mathbf{u}) + \alpha^{-1}(\mathcal{D}(\mathbf{u}) - T) \right\} \tag{2.24}$$

and use a standard optimisation method to solve this problem for some regularisation parameter $\alpha > 0$. Here the function $\mathcal{J}_\alpha(\mathbf{u})$ is referred to as the global objective function, $\mathcal{D}$ is the misfit function or fitting term, and $\mathcal{R}$ is known as the regulariser or the model objective function. The parameter α determines how well the solution should fit the data. As α becomes large the solution fits less well to the data and as α becomes very small, the solution starts to fit noise.

2.5 Discrete PDEs and notation

In several situations one has to solve a discrete version of a continuous partial differential equation (PDE), because the equation cannot be solved analytically or because data is only known at a certain number of discrete locations. A continuous linear boundary value problem in d-dimensions is denoted by

$$L^\Omega u(\mathbf{x}) = f^\Omega(\mathbf{x}) \qquad \text{for } \mathbf{x} = (x_1,\ldots,x_d) \in \Omega, \tag{2.25}$$

$$L^\Gamma u(\mathbf{x}) = f^\Gamma(\mathbf{x}) \qquad \text{for } \mathbf{x} = (x_1,\ldots,x_d) \in \Gamma, \tag{2.26}$$

where Ω is a bounded and open domain in $\mathbb{R}^d$ and $\Gamma = \partial\Omega$ is its boundary.

Example 2.5.1 *One such example would be Poisson's equation in a two-dimensional problem with Dirichlet boundary conditions*

$$-\Delta u(\mathbf{x}) = f^{\Omega}(\mathbf{x}) \qquad in\ \Omega, \tag{2.27}$$

$$u(\mathbf{x}) = f^{\Gamma}(\mathbf{x}) \qquad on\ \Gamma. \tag{2.28}$$

Similarly a continuous nonlinear boundary value problem is defined by

$$N^{\Omega}u(\mathbf{x}) = f^{\Omega}(\mathbf{x}) \qquad \text{for } \mathbf{x} = (x_1, \ldots, x_d) \in \Omega, \tag{2.29}$$

$$L^{\Gamma}u(\mathbf{x}) = f^{\Gamma}(\mathbf{x}) \qquad \text{for } \mathbf{x} = (x_1, \ldots, x_d) \in \Gamma. \tag{2.30}$$

Example 2.5.2 *An example of nonlinear boundary value problems resulting from the well-known variational image denoising model by Rudin, Osher and Fatemi [118] is given as follows:*

$$-\alpha\nabla \cdot \left(\frac{\nabla u(\mathbf{x})}{\sqrt{|\nabla u(\mathbf{x})|^2 + \beta}} \right) + u(\mathbf{x}) = z(\mathbf{x}) \quad in\ \Omega, \tag{2.31}$$

$$\frac{\partial u(\mathbf{x})}{\partial \boldsymbol{n}} = 0 \qquad on\ \Gamma, \tag{2.32}$$

where $\alpha, \beta > 0$. Here $z(\mathbf{x})$ is the noisy image and $u(\mathbf{x})$ is the true image which we wish to recover.

There are various ways that a continuous PDE can be discretised, for example using the finite element method or the finite volume method. For image registration purposes, an image domain $\Omega \subset \mathbb{R}^2$ is usually rectangular and the values of f are known at uniformly distributed points in the domain. Therefore, the natural choice for discretising the domain is to use the *finite difference* method.

Let us consider only two-dimensional problems because it is easy to extend to higher dimensions. Assuming that $\Omega = (a, b) \times (c, d)$ is rectangular we impose a cartesian grid (or mesh) with grid spacing $h = (b-a)/n$ in x-direction and $k = (d-c)/m$ in y-direction. In the so-called *vertex-centered discretisation* grid points are placed at the vertices of the mesh so that there are $(n+1) \times (m+1)$ grid points including points on the boundary with grid point (i, j) located at $(x_i, y_j) = (a+ih, b+jk)$ for $0 \leq i \leq n$ and $0 \leq j \leq m$. In the so-called *cell-centered discretisation* grid points are placed at the centre of the grid cells so that there are $n \times m$ grid points (none lying on the boundary) and grid point (i, j) is located at $(x_i, y_j) = (a + \frac{2i-1}{2}h, c + \frac{2j-1}{2}k)$ for $1 \leq i \leq n$ and $1 \leq j \leq m$. The interior of the discrete grid is denoted by Ω^h and the boundary by Γ^h or $\partial\Omega^h$. Figure 2.2 shows examples of vertex- and cell-centered discretisations of a square domain.

Once the grid is in place the operators in the PDE can be approximated locally using Taylor's series expansion, e.g.

$$u(x+h, y) = u(x, y) + h\frac{\partial u}{\partial x}(x, y) + \frac{h^2}{2}\frac{\partial^2 u}{\partial x^2}(x, y) + \frac{h^3}{3!}\frac{\partial^3 u}{\partial x^3}(x, y) + \frac{h^4}{4!}\frac{\partial 4u}{\partial x^4}(\epsilon_+, y) \tag{2.33}$$

and

$$u(x-h, y) = u(x, y) - h\frac{\partial u}{\partial x}(x, y) + \frac{h^2}{2}\frac{\partial^2 u}{\partial x^2}(x, y) - \frac{h^3}{3!}\frac{\partial^3 u}{\partial x^3}(x, y) + \frac{h^4}{4!}\frac{\partial 4u}{\partial x^4}(\epsilon_-, y) \tag{2.34}$$

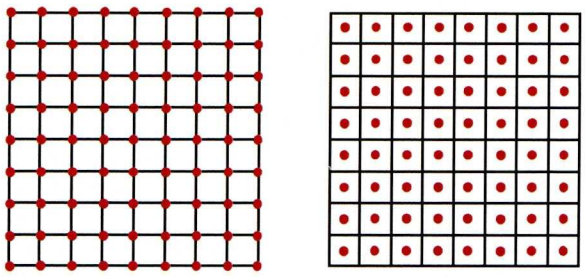


Figure 2.2: Vertex-centered (left) and cell-centered (right) discretisations of a square domain. Filled circles indicate grid points within the square domain.

where $x-h < \epsilon_- < x < \epsilon_+ < x+h$. The operator $\frac{\partial u}{\partial x}$ at the grid point (i,j) can be approximated in 3 ways, the first order forward and backward difference operators defined respectively by

$$\frac{\delta_x^+(u)_{i,j}}{h} = \frac{(u)_{i+1,j} - (u)_{i,j}}{h} \text{ and } \frac{\delta_x^-(u)_{i,j}}{h} = \frac{(u)_{i,j} - (u)_{i-1,j}}{h} \tag{2.35}$$

or the second order central difference approximation

$$\frac{\delta_x^c(u)_{i,j}}{h} = \frac{(u)_{i+1,j} - (u)_{i-1,j}}{2h}, \tag{2.36}$$

where $(u)_{i,j} = u(x_i, y_j)$ is the value of u at the grid point (i,j). Approximations to higher order derivatives can be constructed in a similar way for example a second order approximation to $\frac{\partial^2 u}{\partial x^2}$ at (i,j) is given by

$$\frac{(u)_{i+1,j} - 2(u)_{i,j} + (u)_{i-1,j}}{2h}. \tag{2.37}$$

The discrete analogue of the continuous problem on the discrete domain is denoted by

$$L_h^\Omega u_h(\mathbf{x}) = f_h^\Omega(\mathbf{x}) \qquad \text{for } \mathbf{x} = (x_1, \ldots, x_d) \in \Omega_h, \tag{2.38}$$

$$L_h^\Gamma u_h(\mathbf{x}) = f_h^\Gamma(\mathbf{x}) \qquad \text{for } \mathbf{x} = (x_1, \ldots, x_d) \in \Gamma_h, \tag{2.39}$$

where u_h is a grid function on $\Omega_h \cup \Gamma_h$, L_h^Ω and L_h^Γ are operators on the space of grid functions and f_h^Ω and f_h^Γ are discrete representations of f^Ω and f^Γ. Usually the boundary condition can be eliminated and (2.38) and (2.39) can be written simply as

$$L_h u_h = f_h. \tag{2.40}$$

Example 2.5.3 *Consider Poisson's equation on the unit square with Dirichlet boundary condition*

$$L_h^\Gamma u_h(\mathbf{x}) = f_h^\Gamma(\mathbf{x}) = 0 \qquad \textit{for } \mathbf{x} = (x_1, \ldots, x_d) \in \Gamma_h.$$

Assume that the domain is discretised using a vertex-centered grid with $h = k = 1/n$ then at interior grid points not adjacent to the boundary a second order central difference approximation is given by

$$(L_h u_h)_{i,j} = \frac{4(u)_{i,j} - (u)_{i+1,j} - (u)_{i-1,j} - (u)_{i,j+1} - (u)_{i,j-1}}{h^2} = (f_h)_{i,j}. \tag{2.41}$$

At point adjacent to the right boundary, for example, $(u)_{n+1,j}$ *will be replaced by the boundary value* $(f_h^\Gamma)_{n,j}$*, i.e.*

$$(L_h u_h)_{n,j} = \frac{4(u)_{n,j} - (u)_{n-1,j} - (u)_{n,j+1} - (u)_{n,j-1}}{h^2} = (f_h)_{n,j}. \tag{2.42}$$

Similarly considerations give $L_h u_h$ *at other points adjacent to the boundary, therefore we have* $L_h u_h = f_h$ *where* u_h *is a grid function on the interior grid points only.*

Remark 2.5.1 *For image registration purposes and other image processing applications,* Ω *represents an image domain; see* §3.1 *for the definition of an image. There are different approaches to the discretisation and choice of the image domain* Ω*. Some authors, e.g. [83, 52, 53], use a vertex rather than a cell-centered discretisation of* Ω*. Also the choice of* Ω *is somewhat arbitrary. However, there are two common choices. The first is to take* Ω *to be the unit square whatever the size of the image, i.e.* $\Omega = [0,1] \times [0,1]$ *and* $(h,k) = (1/n, 1/m)$*. The other is to take* Ω *to be such that the grid spacing in each direction is* 1*, e.g. if the image is of size* 512×256 *then* $\Omega = [0,512] \times [0,256]$*. For simplicity, the unit square* $\Omega = [0,1] \times [0,1]$ *with the grid spacing* $h = k = 1/n$ *is adopted and used in all numerical sections throughout this thesis in order to be consistent with the majority of papers that have been seen on this subject, e.g. [47, 57, 62, 76, 83, 91]. Note that if* Ω *is the unit square then the value of* h *is related to image resolution. For example, the value of* h *in the* 256×256 *case will be half what it was in the* 128×128 *case and the discrete image of the* 256×256 *case will be closer to the original or continuous image than that of the* 128×128 *case.*

2.5.1 Stencil notation

Let $p \in \mathbb{Z}^d$ define a grid point on a d-dimensional grid G. In stencil notation the left hand side of the discrete equation $L_h u_h = f_h$ at p is defined by

$$(L_h u_h)_p = \sum_{q \in \mathbb{Z}^d} L_{p,q} (u_h)_{p+q}. \tag{2.43}$$

The stencil entry $L_{p,q}$ is non-zero when $L_h u_h$ at grid point $p \in G$ is dependant on the value of u_h at the grip point $p+q$, the structure of an operator L is defined as all q such that there exists a $p \in G$ such that $L_{p,q}$ is non-zero and is denoted by S_L. The stencil for L_h at p is displayed as an array containing all non-zero $L_{p,q}$, e.g. a typical stencil in 2 dimensions has the form

$$\begin{bmatrix} 0 & L_{p,(0,1)} & 0 \\ L_{p,(-1,0)} & L_{p,(0,0)} & L_{p,(1,0)} \\ 0 & L_{p,(0,-1)} & 0 \end{bmatrix}.$$

Example 2.5.4 *Returning to Example 2.5.3 we saw at grid points not adjacent the boundary we can write*

$$L_h u_h(x,y) = \frac{1}{h^2} \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} u_h(x,y) = f_h(x,y) \tag{2.44}$$

and, for example, at points adjacent to the right boundary

$$L_h u_h(x,y) = \frac{1}{h^2} \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & 0 \\ 0 & -1 & 0 \end{bmatrix} u_h(x,y) = f_h(x,y), \tag{2.45}$$

where u_h now includes points on the right boundary.

2.5.2 Matrix notation

It may sometimes also be useful to write $L_h u_h = f_h$ in terms of matrix notation. This can be done by stacking the grid function u_h into a vector $\mathbf{u}_h$ with the so-called *lexicographical ordering*; u_h is stacked along rows of the grid starting at the bottom left point and ending at the top right. The right hand side vector is stacked in a similar manner into a vector $\mathbf{f}_h$. The discrete linear equation can then be written as $\mathbf{A}_h \mathbf{u}_h = \mathbf{f}_h$; see §2.6.6.

Example 2.5.5 *In the example of Poisson's equation considered previously we see that in a general row l of $\mathbf{A}_h$ one has $a_{l,l} = 4/h^2$ and $a_{l,l-1} = a_{l,l+1} = -1/h^2$ with all other entries in the row zero, with appropriate modifications for boundary points. $h^2\mathbf{A}_h$ is therefore the $(n-1) \times (n-1)$ block tri-diagonal matrix with blocks of size $(n-1) \times (n-1)$ where the off diagonal blocks are the negative identity and the diagonal blocks are tridiagonal with 4 on the diagonal and -1 on the off diagonals.*

2.5.3 Boundary conditions

So far we have only mentioned Dirichlet boundary conditions on vertex-centered grids, we briefly now describe how to deal with Neumann boundary conditions and cell-centered grids.

Neumann boundary conditions for vertex-centered grids

Let us assume that we have a Neumann boundary condition $\frac{\partial u}{\partial \boldsymbol{n}}(x, y) = f^{\Gamma}(x, y)$ on the right boundary of a vertex centered grid. We assume that the discrete equation $L_h^{\Omega} u_h(x, y) = f_h^{\Omega}(x, y)$ extends to the points on the right boundary. The equation at these grid points will involve *ghost* grid points outside the domain. These ghost grid points can be eliminated using the Neumann boundary condition

$$\frac{(u)_{n+1,j} - (u)_{n-1,j}}{2h} = (f^{\Gamma})_{n,j}. \tag{2.46}$$

Example 2.5.6 *If we take the example of Poisson's equation on the unit square then at the right boundary, we have*

$$(L_h u_h)_{n,j} = \frac{4(u)_{n,j} - 2(u)_{n-1,j} - (u)_{n,j+1} - (u)_{n,j-1}}{h^2} = (f_h^{\Omega})_{n,j} + \frac{2}{h}(f_h^{\Gamma})_{n,j}. \tag{2.47}$$

Neumann boundary conditions for cell-centered grids

In the case of a cell-centered grid we have no points on the boundary, so in general the equation at interior points which are adjacent to the boundary will involve ghost points outside of the domain, which need to be eliminated using the boundary condition. If we have a Neumann boundary condition at this right boundary, for example, we can write it as

$$\frac{(u)_{n+1,j} - (u)_{n,j}}{h} = (f^{\Gamma})_{n,j}. \tag{2.48}$$

2.5.4 Nonlinear equations

Nonlinear PDEs are treated in much the same way as linear equations, the various operators in the equation are approximated locally on a discrete grid using the finite difference method. The discrete nonlinear equation is denoted by

$$N_h^\Omega u(\mathbf{x}) = f_h^\Omega(\mathbf{x}) \qquad \text{for } \mathbf{x} = (x_1, \ldots, x_d) \in \Omega_h, \tag{2.49}$$

$$L_h^\Gamma u(\mathbf{x}) = f_h^\Gamma(\mathbf{x}) \qquad \text{for } \mathbf{x} = (x_1, \ldots, x_d) \in \Gamma_h. \tag{2.50}$$

Similarly the boundary conditions are usually eliminated and then the discrete nonlinear equation can be written simply as

$$N_h(u_h) = f_h. \tag{2.51}$$

It may be possible to write the nonlinear equation in a matrix notation, e.g. $\mathbf{A}_h(\mathbf{u}_h)\mathbf{u}_h = \mathbf{f}_h$.

For more on finite difference methods for PDEs see for example [103, 128].

2.6 Basic iterative methods

This section introduces a class of iterative methods for solving a general linear system of equations

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{2.52}$$

where $\mathbf{x} \in \mathbb{R}^N$ and $\mathbf{A}$ is a matrix of size $N \times N$. These iterative methods start with some initial approximation $\mathbf{x}^{(0)}$ and then generate a sequence $\{\mathbf{x}^{(k)}\}_{k=1}^{\infty}$ via the relation

$$\mathbf{x}^{(k)} = \mathbf{T}\mathbf{x}^{(k-1)} + \mathbf{c}. \tag{2.53}$$

The relation matrix $\mathbf{T}$ and the vector $\mathbf{c}$ come from a splitting $\mathbf{A} = \mathbf{M} - \mathbf{N}$ of the matrix $\mathbf{A}$ where $\mathbf{M}$ is nonsingular. With this splitting the original system (2.52) can then be written as

$$\mathbf{A}\mathbf{x} = (\mathbf{M} - \mathbf{N})\mathbf{x} = \mathbf{b}$$

or

$$\mathbf{x} = (\mathbf{M}^{-1}\mathbf{N})\mathbf{x} + \mathbf{M}^{-1}\mathbf{b} = \mathbf{T}\mathbf{x} + \mathbf{c} \tag{2.54}$$

where $\mathbf{T} = \mathbf{M}^{-1}\mathbf{N}$ and $\mathbf{c} = \mathbf{M}^{-1}\mathbf{b}$.

Each application of an iterative methods which updates $\mathbf{x}^{(k-1)}$ to $\mathbf{x}^{(k)}$ is known as an iteration or a relaxation sweep.

2.6.1 Jacobi method

The Jacobi method consists of solving the ith equation of $\mathbf{A}\mathbf{x} = \mathbf{b}$ for x_i to get

$$x_i = \sum_{\substack{j=1 \\ j \neq i}}^{N} \left(\frac{-a_{ij}x_j}{a_{ii}} \right) + \frac{b_i}{a_{ii}} \quad \text{for } i = 1, \ldots, N. \tag{2.55}$$

Then given $\mathbf{x}^{(k-1)}$ for $k \geq 1$, $\mathbf{x}^{(k)}$ is generated by

$$x_i^{(k)} = \sum_{\substack{j=1 \\ j \neq i}}^{N} \left(\frac{-a_{ij} x_j^{(k-1)}}{a_{ii}} \right) + \frac{b_i}{a_{ii}} \text{ for } i = 1, \ldots, N. \tag{2.56}$$

Note that we require $a_{ii} \neq 0$ for each $i = 1, \ldots, N$. If one or more $a_{ii} = 0$ and the system is nonsingular then a reordering can be performed so that no a_{ii} equals 0. To write $\mathbf{Ax} = \mathbf{b}$ in the form $\mathbf{x} = \mathbf{Tx} + \mathbf{c}$ we write $\mathbf{A}$ as $\mathbf{A} = \mathbf{D} - \mathbf{L} - \mathbf{U}$ where $\mathbf{D}$ is a diagonal matrix whose diagonal is the same as that of $\mathbf{A}$, $-\mathbf{L}$ is the strictly lower triangle part of $\mathbf{A}$ and $-\mathbf{U}$ is the strictly upper triangle part of $\mathbf{A}$. Therefore, we have

$$\mathbf{Ax} = (\mathbf{D} - \mathbf{L} - \mathbf{U})\mathbf{x} = \mathbf{b}$$

or

$$\mathbf{x} = \mathbf{D}^{-1}(\mathbf{L} + \mathbf{U})\mathbf{x} + \mathbf{D}^{-1}\mathbf{b}, \tag{2.57}$$

i.e. we use matrix splitting $\mathbf{A} = \mathbf{M} - \mathbf{N}$ where $\mathbf{M} = \mathbf{D}$ and $\mathbf{N} = \mathbf{L} + \mathbf{U}$. The matrix form of the Jacobi method is then given by

$$\mathbf{x}^{(k)} = \mathbf{T}_J \mathbf{x}^{(k-1)} + \mathbf{c}_J, \tag{2.58}$$

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

Algorithm for Jacobi method

The Jacobi algorithm for finding an approximate solution to $\mathbf{Ax} = \mathbf{b}$ given an initial approximation $\mathbf{x}^{(0)}$ is given below (Algorithm 2.6.1). A maximum number of iterations $IMAX$ to be performed and a tolerance $\epsilon > 0$ to terminate the algorithm must be specified.

Algorithm 2.6.1 (Jacobi Method)

$$[\mathbf{x}] \leftarrow JAC(\mathbf{A}, \mathbf{b}, x^{(0)}, IMAX, \epsilon)$$

---

1. Set $k = 1$, $N = size(\mathbf{x}^{(0)})$, done = False
2. While done = False do steps 3-4
3. For $i = 1, \ldots, N$,
   - Set $x_i^{(k)} = \sum_{\substack{j=1 \\ j \neq i}}^{N} \left(\frac{-a_{ij} x_j^{(k-1)}}{a_{ii}} \right) + \frac{b_i}{a_{ii}}$
4. If $||\mathbf{b} - \mathbf{Ax}^{(k)}||_2 < \epsilon$ OR $||\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}||_2 < \epsilon$ OR $k \geq IMAX$,
   - Set done = True and $\mathbf{x} = \mathbf{x}^{(k)}$

   else
   - Set $k = k + 1$

   end

Weighted Jacobi method

In the weighted Jacobi method, given the current approximation $\mathbf{x}^{(k-1)}$ the new Jacobi steps are computed using

$$\overline{x}_i^{(k)} = \sum_{\substack{j=1 \\ j \neq i}}^{N} \left(\frac{-a_{ij} x_j^{(k-1)}}{a_{ii}} \right) + \frac{b_i}{a_{ii}} \text{ for } i = 1, \ldots, N \tag{2.59}$$

as before, however $\overline{\mathbf{x}}^{(k)}$ is now just an intermediate value. The new approximation $\mathbf{x}^{(k)}$ is given by

$$\mathbf{x}^{(k)} = (1-\omega)\mathbf{x}^{(k-1)} + \omega\overline{\mathbf{x}}^{(k)} \tag{2.60}$$

where $0 \leq \omega < 2$ is a weighted factor to be chosen. Of course when $\omega = 1$ we have the original Jacobi method. In matrix form the weighted Jacobi method is given by

$$\mathbf{x}^{(k)} = ((1-\omega)\mathbf{I} + \omega\mathbf{T}_J)\mathbf{x}^{(k-1)} + \omega\mathbf{c}_J \tag{2.61}$$

which is equivalent to

$$\mathbf{x}^{(k)} = \mathbf{T}_{J,\omega}\mathbf{x}^{(k-1)} + \mathbf{c}_{J,\omega} \tag{2.62}$$

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

2.6.2 Gauss-Seidel method

When computing $x_i^{(k)}$ in the Jacobi method we have already computed $x_1^{(k)}$, $x_2^{(k)}$, ..., $x_{i-1}^{(k)}$ which should be better approximations to x_1, x_2, ..., x_{i-1} than $x_1^{(k-1)}$, $x_2^{(k-1)}$, ..., $x_{i-1}^{(k-1)}$. Therefore the Jacobi method should be improved if we rewrite the equation for $x_i^{(k)}$ as

$$x_i^{(k)} = \frac{-\sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^{N} a_{ij}x_j^{(k-1)} + b_i}{a_{ii}} \quad \text{for } i = 1, \ldots, N. \tag{2.63}$$

This is known as the Gauss-Seidel method. Rewriting the above equation as

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

leads to the matrix form of the Gauss-Seidel method as

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

or equivalently

$$\mathbf{x}^{(k)} = \mathbf{T}_{GS}\mathbf{x}^{(k-1)} + \mathbf{c}_{GS} \tag{2.64}$$

where $\mathbf{T}_{GS} = (\mathbf{D}-\mathbf{L})^{-1}\mathbf{U}$ and $\mathbf{c}_{GS} = (\mathbf{D}-\mathbf{L})^{-1}\mathbf{b}$. Gauss-Seidel is therefore based on a matrix splitting with $\mathbf{M} = \mathbf{D}-\mathbf{L}$ and $\mathbf{N} = \mathbf{U}$.

Algorithm for Gauss-Seidel method

The algorithm for the Gauss-Seidel method is the same as the algorithm for the Jacobi method, except that step 3 is replaced by (2.63).

2.6.3 SOR method

In the successive over relaxation (SOR) method given $x_i^{(k-1)}$ the intermediate values $\overline{x}_i^{(k)}$ are computed using Gauss-Seidel and these values are used to evaluate $\mathbf{x}^{(k)}$ as follows:$\mathbf{A}_{1s}$

$$\mathbf{x}^{(k)} = (1-\omega)\mathbf{x}^{(k-1)} + \omega\overline{\mathbf{x}}^{(k)} \tag{2.65}$$

where $\omega > 0$ is known as the relaxation parameter. If $0 < \omega < 1$ this iterative scheme is called under-relaxation and is used to obtain convergence when the Gauss-Seidel does not converge. If $\omega > 1$ it is called over-relaxation and it is used to accelerate convergence of the system which are converge by the Gauss-Seidel method. SOR is based on the matrix splitting

$$\omega\mathbf{A} = (\mathbf{D} - \omega\mathbf{L}) - (\omega\mathbf{U} + (1-\omega)\mathbf{D})$$

and can be defined by the recurrence

$$\mathbf{x}^{(k)} = \mathbf{T}_{SOR}\mathbf{x}^{(k-1)} + \mathbf{c}_{SOR} \tag{2.66}$$

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

2.6.4 Block methods

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

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

Then $\mathbf{Ax} = \mathbf{b}$ can be written in the block form as follows:

$$\underbrace{\begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \ldots & \mathbf{A}_{1s} \\ \mathbf{A}_{21} & \mathbf{A}_{22} & \ldots & \mathbf{A}_{2s} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{A}_{s1} & \mathbf{A}_{s2} & \ldots & \mathbf{A}_{ss} \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_s \end{pmatrix}}_{\mathbf{x}} = \underbrace{\begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \vdots \\ \mathbf{b}_s \end{pmatrix}}_{\mathbf{b}}. \tag{2.67}$$

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 $\mathbf{b}_p$ is of size N_p. Thus, for any vector $\mathbf{x}$ partitioned as in (2.67),

$$(\mathbf{Ax})_i = \sum_{j=1}^{s} \mathbf{A}_{ij}\mathbf{x}_j$$

in which $(y)_i$ denotes the i-th component of the vector i according to the above partitioning. Assuming that the diagonal blocks are nonsingular the Jacobi and Gauss-Seidel methods can easily extended to the block level. In the block Jacobi method for $i = 1, \ldots, s$, $\mathbf{x}_i$ is updated as follows:

$$\mathbf{x}_i^{(k)} = \mathbf{A}_{ii}^{-1}\left(\sum_{\substack{j=1 \\ j\neq i}}^{N} - \mathbf{A}_{ij}\mathbf{x}_j^{(k-1)} + \mathbf{b}_i\right). \tag{2.68}$$

Similarly in the block Gauss-Seidel method $\mathbf{x}_i$ is updated as

$$\mathbf{x}_i^{(k)} = \mathbf{A}_{ii}^{-1}\left(\sum_{j=1}^{i-1} - \mathbf{A}_{ij}\mathbf{x}_j^{(k)} + \sum_{j=i+1}^{s} - \mathbf{A}_{ij}\mathbf{x}_j^{(k)} + \mathbf{b}_i\right). \tag{2.69}$$

If we define $\mathbf{D}_B$, $\mathbf{U}_B$ and $\mathbf{L}_B$ for the splitting $\mathbf{A} = \mathbf{D}_B - \mathbf{L}_B - \mathbf{U}_B$ as block analogues of $\mathbf{D}$, $\mathbf{U}$ and $\mathbf{L}$, i.e.

$$\mathbf{D}_B = \begin{bmatrix} \mathbf{A}_{11} & & & \\ & \mathbf{A}_{22} & & \\ & & \ddots & \\ & & & \mathbf{A}_{ss} \end{bmatrix},$$

$$\mathbf{U}_B = -\begin{bmatrix} \mathbf{0} & & & \\ \mathbf{A}_{21} & \mathbf{0} & & \\ \vdots & \vdots & \ddots & \\ \mathbf{A}_{s1} & \mathbf{A}_{s2} & \dots & \mathbf{0} \end{bmatrix}, \qquad \mathbf{L}_B = -\begin{bmatrix} \mathbf{0} & \mathbf{A}_{12} & \dots & \mathbf{A}_{1s} \\ & \mathbf{0} & \dots & \mathbf{A}_{2s} \\ & & \ddots & \vdots \\ & & & \mathbf{0} \end{bmatrix},$$

then (2.68) can be re-written as

$$\mathbf{x}_i^{(k)} = \mathbf{A}_{ii}^{-1}((\mathbf{L}_B + \mathbf{U}_B)\mathbf{x}^{(k-1)})_i + \mathbf{A}_{ii}^{-1}\mathbf{b}_i, \qquad i = 1, \dots, s$$

which leads to the block Jacobi method described by the recurrence

$$\mathbf{x}^{(k)} = \mathbf{D}_B^{-1}\left(\mathbf{L}_B + \mathbf{U}_B\right)\mathbf{x}^{(k-1)} + \mathbf{D}_B^{-1}\mathbf{b} \tag{2.70}$$

and similarly Block Gauss-Seidel by the recurrence

$$\mathbf{x}^{(k)} = (\mathbf{D}_B - \mathbf{L}_B)^{-1}\mathbf{U}_B\mathbf{x}^{(k-1)} + (\mathbf{D}_B - \mathbf{L}_B)^{-1}\mathbf{b}. \tag{2.71}$$

Note that an important difference between block relaxation schemes and point relaxation schemes is that now $\mathbf{A}_{ii}$ is a matrix instead of a scalar a_{ii}. As a result, solving linear system with $\mathbf{A}_{ii}$ in (2.68) and (2.69) for updating $\mathbf{x}_i$ may be much more expensive because the matrix inversion of the diagonal block $\mathbf{A}_{ii}$ is required instead of the inverse of the scalar a_{ii}. Obviously the larger the vectors $\mathbf{x}_i$ are, the more expensive each step of the methods is likely to be, on the other hand the payoff may be faster convergence of the iterative method. Nevertheless, the number of iterations required to achieve convergence often decreases rapidly because they update the whole set of components at each time [121].

2.6.5 Convergence

The methods considered in this section all define a sequence of iterates $\mathbf{x}^{(k)} = \mathbf{T}\mathbf{x}^{(k-1)} + \mathbf{c}$, which upon convergence produce a solution of the original system $\mathbf{A}\mathbf{x} = \mathbf{b}$. In the following it is shown that the iteration $\mathbf{x}^{(k)} = \mathbf{T}\mathbf{x}^{(k-1)} + \mathbf{c}$ converges if and only if the spectral radius of $\mathbf{T}$ is less than one. First the definition of a convergence matrix is required.

Definition 2.6.1 (*Convergence Matrix*) *A square matrix* $\mathbf{A}$ *is said to be convergent if* $\lim_{k\to\infty} \mathbf{A}^k = 0$.

The following theorem, the proof of which can be found in [121], is also required.

Theorem 2.6.1 *A matrix* $\mathbf{A}$ *is convergent if and only if* $\rho(\mathbf{A}) < 1$, *where* $\rho(\mathbf{A})$ *is the spectral radius of* $\mathbf{A}$.

Finally we also need the following theorem.

Theorem 2.6.2 *For any initial guess solution* $\mathbf{x}^{(0)} \in \mathbb{R}^N$ *the sequence* $\left\{\mathbf{x}^{(k)}\right\}_{k=1}^{\infty}$ *defined by* $\mathbf{x}^{(k)} = \mathbf{T}\mathbf{x}^{(k-1)} + \mathbf{c}$ *for all* $k \geq 1$ *converges to the unique solution of* $\mathbf{x} = \mathbf{T}\mathbf{x} + \mathbf{c}$ *if and only if* $\rho(\mathbf{T}) < 1$.

Below some useful theorems on convergence are stated without proofs.

Theorem 2.6.3 *If a matrix* $\mathbf{A}$ *has positive diagonal entries and all other entries are negative or zero then only one of the following statements holds*

1. $0 \leq \rho(\mathbf{T}_{GS}) < \rho(\mathbf{T}_J) < 1$
2. $1 < \rho(\mathbf{T}_J) < \rho(\mathbf{T}_{GS})$
3. $\rho(\mathbf{T}_J) = \rho(\mathbf{T}_{GS}) = 0$
4. $\rho(\mathbf{T}_J) = \rho(\mathbf{T}_{GS}) = 1$

where $\mathbf{T}_J$ *and* $\mathbf{T}_{GS}$ *are the iteration matrices for Jacobi and Gauss-Seidel respectively.*

This theorem implies that for such matrices if one of Jacobi or Gauss-Seidel converges then so does the other and similarly divergence of one implies divergence of the other. If both converge then Gauss-Seidel converges faster then Jacobi. For the next theorems we need to define a regular splitting of $\mathbf{A}$.

Definition 2.6.2 *(Regular Splitting)* $\mathbf{A} = \mathbf{M} - \mathbf{N}$ *is a regular splitting of* $\mathbf{A}$ *if* $\mathbf{M}$ *is nonsingular and* $\mathbf{M}^{-1}$ *and* $\mathbf{N}$ *are nonnegative.*

Theorem 2.6.4 *If* $\mathbf{M}$ *and* $\mathbf{N}$ *are regular splitting of* $\mathbf{A}$ *and* $\mathbf{T} = \mathbf{M}^{-1}\mathbf{N}$ *then* $\rho(\mathbf{T}) < 1$ *if and only if* $\mathbf{A}$ *is nonsingular and* $\mathbf{A}^{-1}$ *is nonnegative.*

Theorem 2.6.5 *If all the diagonal elements of* $\mathbf{A}$ *are non-zero then* $\rho(\mathbf{T}_{SOR}) \geq |\omega - 1|$ *and hence SOR converge only when* $0 < \omega < 2$.

Theorem 2.6.6 *If* $\mathbf{A}$ *is positive definite, i.e.* $\mathbf{x}^\top \mathbf{A}\mathbf{x} > 0$ *for all* $\mathbf{x}$, *and* $0 < \omega < 2$ *then the SOR method converge for any initial guess* $\mathbf{x}^{(0)}$.

Theorem 2.6.7 *If* $\mathbf{A}$ *is positive definite and tri-diagonal then* $\rho(\mathbf{T}_{GS}) = \rho(\mathbf{T}_J)^2$ *and the optimal* ω *for SOR is*

$$\omega = \frac{2}{1 + \sqrt{1 - \rho(\mathbf{T}_J)^2}}$$

for which $\rho(\mathbf{T}_{SOR}) = \omega - 1$.

2.6.6 Numerical implementation

If we have a system of equations $\mathbf{A}\mathbf{u} = \mathbf{f}$ arising from the discretisation of a PDE using a 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 $\mathbf{u}$ will typically involve just a few other entries. To illustrate this the numerical implementation of Jacobi and Gauss-Seidel methods is outlined for the case of Poisson's equation with Dirichlet boundary conditions on the unit square introduced in §2.6. For ease of presentation a grid function will be used.

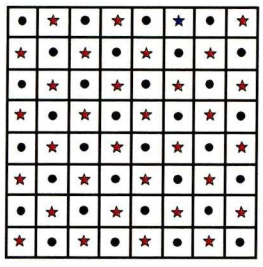


Figure 2.3: Red-Black ordering of grid points: red points are shown as starts and black are shown as circles.

Jacobi method

In the weighted Jacobi method if a grid point (i, j) is not adjacent to the boundary then $(u)_{i,j}$ is updated as follows:

$$(u)_{i,j}^{(k)} = (1-\omega)(u)_{i,j}^{(k-1)} + \omega \left[\frac{h^2(f)_{i,j} + (u)_{i+1,j}^{(k-1)} + (u)_{i-1,j}^{(k-1)} + (u)_{i,j+1}^{(k-1)} + (u)_{i,j-1}^{(k-1)}}{4} \right], \quad (2.72)$$

where for example $(u)_{i+1,j}^{(k-1)}$ is the entry of the previous approximation $(u^h)^{(k-1)}$ corresponding to the grid point $(i+1, j)$. For points adjacent to the boundary appropriate modifications to (2.72) should be made.

Gauss-Seidel method

Unlike in the Jacobi method the order in which entries of u^h are updated is significant when using the Gauss-Seidel method. Two different ordering schemes (corresponding to two different ways of stacking u^h into a vector) for Gauss-Seidel are outlined below.

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 $(n-1, m-1)$ updated last. A Gauss-Seidel scheme used with lexicographic ordering is denoted GS-LEX and the entry u^h corresponding to grid point (i, j) (not adjacent to the boundary) is updated as follows:

$$(u)_{i,j}^{(k)} = \frac{h^2(f)_{i,j} + (u)_{i+1,j}^{(k-1)} + (u)_{i-1,j}^{(k+1)} + (u)_{i,j+1}^{(k-1)} + (u)_{i,j-1}^{(k+1)}}{4}. \quad (2.73)$$

Note that because of the lexicographic ordering entries corresponding to points to the left of and below (i, j) have already been updated whereas entries corresponding to points to the right of and above (i, j) have not.

Red-Black ordering

When a red-black ordering of the grid points is used the grid is coloured in a checkerboard fashion as shown in Figure 2.3, 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 GS-RB. Entries of u^h corresponding to red grid points are updated by

$$(u)_{i,j}^{(k)} = \frac{h^2(f)_{i,j} + (u)_{i+1,j}^{(k-1)} + (u)_{i-1,j}^{(k-1)} + (u)_{i,j+1}^{(k-1)} + (u)_{i,j-1}^{(k-1)}}{4} \tag{2.74}$$

and then entries corresponding to black grid points are updated by

$$(u)_{i,j}^{(k)} = \frac{h^2(f)_{i,j} + (u)_{i+1,j}^{(k)} + (u)_{i-1,j}^{(k)} + (u)_{i,j+1}^{(k)} + (u)_{i,j-1}^{(k)}}{4}. \tag{2.75}$$

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 correspondingly 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 intimal guess.

Line relaxation

If u^h is stacked into a vector $\mathbf{u}$ lexicographically and the vector $\mathbf{u}$ is divided into $(n-1)$ subvectors where each of them is of size $(n-1)$, then the subvector $\mathbf{u}_l$ will contain 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. For example, the Gauss-Seidel updating of $\mathbf{u}_l$ is done as follows:

$$(\mathbf{u}_l)^{(k)} = \mathbf{A}_{ll}^{-1}(h^2\mathbf{f}_l + (\mathbf{u}_{l-1})^{(k)} + (\mathbf{u}_{l+1})^{(k)}), \tag{2.76}$$

where $\mathbf{A}_{ll}$ is a tri-diagonal 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 pointwise relaxation for line Gauss-Seidel is the zebra line relaxation; here either rows or columns of the grid are coloured 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.

2.6.7 Local nonlinear relaxation methods

If we have a discrete nonlinear PDE $N^h(u^h) = f^h$ on a grid Ω_h which has in total N grid points then we have in general a system of nonlinear equations:

$$W_i(u_1, u_2, \ldots, u_N) = 0, \quad i = 1, 2, \ldots, N. \tag{2.77}$$

Analogous to the linear case a nonlinear Jacobi iteration involves solving the ith equation for the ih unknown

$$W_i(u_1^{(k)}, u_2^{(k)}, \ldots, u_{i-1}^{(k)}, u_i^{(k+1)}, u_{i+1}^{(k)}, \ldots, u_N^{(k)}) = 0, \tag{2.78}$$

where k denotes the current approximation, $k+1$ denotes the new approximation and we start with some initial guess $u^{(0)}$. Similarly a nonlinear Gauss-Seidel iteration is given by

$$W_i(u_1^{(k+1)}, u_2^{(k+1)}, \ldots, u_{i-1}^{(k+1)}, u_i^{(k+1)}, u_{i+1}^{(k)}, \ldots, u_N^{(k)}) = 0, \quad i = 1, 2, \ldots, N, \tag{2.79}$$

where of course $u_1, \ldots, u_{i-1}$ are known before u_i is updated. Both these methods will involve solving a nonlinear equation in one unknown to update each u_i.

For finding the root α of a scalar equation $W(u) = 0$, the standard Newton iteration is defined by

$$u^{(m+1)} = u^{(m)} - W(u^{(m)})/W'(u^{(m)}) \quad m = 1, 2, \ldots,$$

where $u^{(0)}$ is the initial iterate and $W'(\alpha) \neq 0$. Using this scheme to solve for $u_i^{(k+1)}$ in (2.78) and (2.79), it follows that

$$u_i^{(k+1,m+1)} = u_i^{(k+1,m)} - \frac{W_i(u_i^{(k+1,m)})}{C(u_i^{(k+1,m)})},$$

where

$$C(u_i^{(k+1,m)}) = \frac{\partial W_i(u_i^{(k+1,m)})}{\partial u_i}$$

and $u_i^{(k+1,0)} = u_i^{(k)}$ is the the initial guess. Here the dependence on u_j $(j \neq i)$ has been suppressed for notational convenience. For the case (2.79) we imply, for instance, that

$$W_i(u_i^{(k,m)}) \equiv W_i(u_1^{(k+1)}, u_2^{(k+1)}, \ldots, u_{i-1}^{(k+1)}, u_i^{(k,m)}, u_{i+1}^{(k)}, \ldots, u_N^{(k)}).$$

We note that using one step of Newton's method, it yields

$$u_i^{(k+1)} = u_i^{(k)} - \frac{W_i(u_i^{(k)})}{C(u_i^{(k)})}, \tag{2.80}$$

where

$$C(u_i^{(k)}) = \frac{\partial W_i(u_i^{(k)})}{\partial u_i}. \tag{2.81}$$

The resulting iteration are known as Jacobi-Newton and Gauss-Seidel-Newton. Refer to [21] and [134, p. 151].

In the case where we have a semi-linear system of equations so that at each grid point we have

$$a_1 u_1 + \ldots + a_N u_N + W_i(u_1, u_2, \ldots, u_N) = 0, \tag{2.82}$$

where W is a nonlinear equation, the Jacobi-Newton iteration is performed by substituting in $u_j^{(k)}$ for $j \neq i$ and then replacing $W_i(u_i{}^{(k+1)})$ by

$$W_i(u_i^{(k)}) = C(u_i^{(k)})(u_i^{(k+1)} - (u_i^{(k)}).$$

u_i is then updated by

$$u_i^{(k+1)} = -\frac{1}{a_i + C(u_i^{(k)})}\left[\left(a_1 u_1^{(k)} + \ldots + a_{i-1}u_{i-1}^{(k)} + a_{i+1}u_{i+1}^{(k)} + \ldots + a_N u_N^{(k)}\right)\right.$$
$$\left. + W_i(u_i^{(k)}) - C(u_i^{(k)})u_i^{(k)}\right]. \tag{2.83}$$

Alternatively we can simply substitute $u_i^{(k)}$ into W_i as follows:

$$u_i^{(k+1)} = -\frac{1}{a_i}\left[\left(a_1 u_1^{(k)} + \ldots + a_{i-1}u_{i-1}^{(k)} + a_{i+1}u_{i+1}^{(k)} + \ldots + a_N u_N^{(k)}\right)\right.$$
$$\left. + W_i(u_1^{(k)}, \ldots, u_i^{(k)}, \ldots, u_p^{(k)})\right], \tag{2.84}$$

which is known as the Jacobi-Picard iteration. A Gauss-Seidel-Picard iteration can be defined in a similar way.

Remark 2.6.1 *As in the case of linear PDEs, we expect that a discrete nonlinear PDE at a particular point will be defined in terms of u at that grid point and a small number of neighbouring points.*

2.7 Multigrid methods

Multigrid (MG) methods, first developed by A. Brandt in the 1970s, have been proven to be fast efficient solvers for a wide range of linear and nonlinear elliptic PDEs discretised on structured and unstructured grids in several applications. The basic idea of a MG method is to smooth high frequency components of the error of the solution in the Fourier modes by performing a few steps with a so-called *smoother* (an iterative relaxation technique like the ones discussed in the previous section) such that a smooth error term can be well represented and approximated on a coarser grid. After a linear or nonlinear residual equation has been solved accurately on the coarse grid, a coarse-grid correction is interpolated back to the fine grid and used to correct the fine grid approximation. Finally, the smoother is performed again in order to remove some new high frequency components of the error introduced by the interpolation. Recursive use of the idea leads to a MG method; see §2.7.5.

In the following sections, the basic principles including the main components of MG methods are briefly described.

2.7.1 Basic principles of multigrid methods

The two basic principles of MG methods are *error smoothing* and *coarse-grid correction.*

Error smoothing

Several basic relaxation techniques are slow to converge like the ones discussed in the previous section when they are used to solve discrete elliptic PDEs which are discretised on cartesian grids. However they do (if applied appropriately) posses what is known as the *smoothing property.* In Fourier modes, these iterative techniques eliminate rapidly the high frequency

components of the error of the solution but may not be effective at removing the low frequency components of the error (leading to slow convergence for being stand-alone solvers). However, a smooth quality can be well represented and approximated on a coarser grid, which leads to the second principle on which MG methods are built.

Coarse-grid correction

Consider a linear system

$$\mathbf{Au} = \mathbf{f}. \tag{2.85}$$

Let $\mathbf{v}$ be an approximation to the exact solution $\mathbf{u}$. Then the error of the solution is defined by:

$$\mathbf{e} = \mathbf{u} - \mathbf{v}. \tag{2.86}$$

Applying $\mathbf{A}$ to both sides of (2.86) leads to the so-called *residual* or *defect equation* given by

$$\mathbf{Ae} = \mathbf{A}(\mathbf{u} - \mathbf{v}) = \mathbf{Au} - \mathbf{Av} = \mathbf{f} - \mathbf{Av} = \mathbf{r}. \tag{2.87}$$

From here we see that if the residual equation (2.87) is solved exactly, then one can obtain $\mathbf{u}$ through $\mathbf{u} = \mathbf{v} + \mathbf{e}$. However, solving the residual equation (2.87) is as expensive as solving the original equation (2.85). To tackle this problem we first replace $\mathbf{A}$ by an appropriate and simpler approximation $\widehat{\mathbf{A}}$ in such a way that the approximation of the error $\widehat{\mathbf{e}}$ can be cheaply computed and used to correct $\mathbf{v}$, and then repeat the process until it reaches the convergence.

To illustrate precisely the main idea of MG methods, let us now focus on the linear system $L_h u_h = f_h$ resulting from an elliptic PDE on the fine grid Ω^h with grid spacing (h, k). Let v_h be an approximation solution computed by performing a few steps with a smoother (**pre-smoothing step**) on the fine-grid problem. Then, the residual equation is given by

$$L_h e_h = f_h - L_h v_h = r_h, \tag{2.88}$$

where $e_h = u_h - v_h$ is the error of the solution, which should not be computed directly on the fine grid. Since high frequency components of the error in pre-smoothing step have already been removed by the smoother, we can transfer the following residual equation to the coarse grid Ω^H with grid spacing (H, K) as follows

$$L_h e_h = r_h \rightarrow L_H e_H = I_h^H r_h = r_H. \tag{2.89}$$

Here L_H is assumed to be an appropriate approximation of L_h on the coarse grid, which is usually the original operator discretised on Ω^H, and I_h^H and I_H^h are two transfer operators capable to convert vectors between Ω^h and Ω^H. After the residual equation system (2.89) on the coarse grid have been solved exactly with a method of our choice, the coarse-grid correction e_H is then interpolated back to the fine grid one e_h by the interpolation operator I_H^h (i.e. $e_h = I_H^h e_H$) that can now be used for updating the approximated solution v_h of the original linear system on the fine grid by $v_h^{new} = v_h + e_h$ (**coarse-grid correction step**). The last

step for a MG method is to perform the smoother again to remove high frequency parts of the interpolated error (**post-smoothing step**).

Overall the MG procedure, which is discussed above and also known as the *two–grid correction scheme*, can be summarised as follows

1. **Pre-smoothing step:**

   - Solve approximately $L_h u_h = f_h$ by performing a few steps with a smoother to obtain the approximation v_h on the fine grid Ω^h

2. **Coarse-grid correction step:**

   - Compute the fine grid residual $r_h = f_h - L_h v_h$ and transport to the coarse grid Ω^H by $r_H = I_h^H r_h$
   - Solve accurately (or exactly) the residual equation $L_H e_H = r_H$ on the coarse grid Ω^H
   - Interpolate the error from the coarse grid to the fine grid by $e_h = I_H^h e_H$
   - Correct the fine grid approximation by $v_h = v_h + e_h$

3. **Post-smoothing step:**

   - Solve approximately $L_h u_h = f_h$ with the initial guess v_h by performing a few steps with the smoother on the fine grid Ω^h

Remark 2.7.1 *A similar MG procedure can be used for a nonlinear system $N_h u_h = f_h$ by using the nonlinear residual equation, which is introduced later in* §2.7.8.

Clearly this MG procedure will only be effective if the error e_h can be well represented and approximated on a coarser grid, i.e. it is *smooth*. The combination of iterative methods which are slow to converge but nevertheless smooth the error, with coarse grid correction is the main idea behind MG methods.

The next three subsections discuss more precisely what is meant by coarse grids, restriction and interpolation operators, and smoothing properties.

2.7.2 Coarsening

MG methods are based on the use of coarse grids to accelerate iterative methods. This section describes in more detail what is meant by a coarse grid. Here we assume that we have a cartesian grid Ω^h with grid spacing (h, k) called the fine grid and construct a coarse grid Ω^H with grid spacing (H, K).

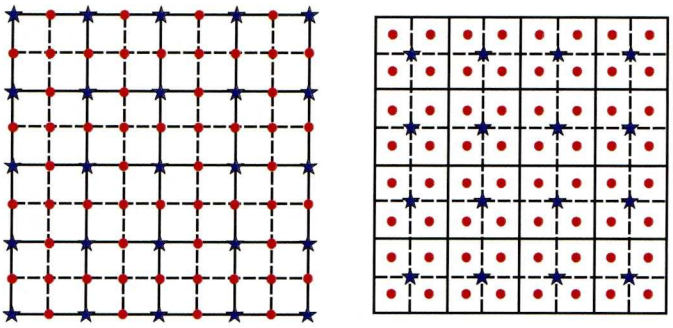


Figure 2.4: Fine and coarse grids in the vertex-centered case (left) and the cell-centered case (right). Coarse grid lines are full, additional fine grid lines are dashed. Circles are fine grid points, stars are coarse grid points in the cell-centered case and points which are both coarse and fine in the vertex centered case.

Standard coarsening

Standard coarsening is the simplest and most frequent way to construct a coarse grid Ω^H by doubling the grid spacing in all directions, i.e. $(H, K) = (2h, 2k)$. In the case of a vertex-centered grid, if Ω^h has $(n+1) \times (m+1)$ grid points including boundary points then Ω^H will have $(n/2+1) \times (m/2+1)$ grid points including boundary points and the coarse grid points will be a subset of the set of the fine grid points. For example the coarse grid point $(1, 1)$ located at $(a + 2h, c + 2k)$ is the same as the fine grid point $(2, 2)$. On the other hand, for a cell-centered discretisation, if the fine grid has $n \times m$ grid points then the coarse grid has $n/2 \times m/2$ grid points. It is different from the vertex-centered case that the coarse grid points will not coincide with fine grid points; see Figure 2.4 for an example of fine and coarse vertex-centered and cell-centered grids.

Other coarsening

Other types of coarsening aside from standard coarsening can be used, for example the grid spacing can be doubled in just one direction e.g. $(H, K) = (h, 2k)$ this is known as semi-coarsening. Semi-coarsening is used in anisotropic problems where pointwise smoothers smooth the error in only one direction.

2.7.3 Transfer operators

As is well known, in addition to a smoother, transfer operators are also the main MG components, which are used to transfer grid functions between different grids. Transferring grid functions from a fine to a coarse grid is known as *restriction*. On the other hand, transferring grid functions from a coarse to a fine grid is called *interpolation* or *prolongation*. In the following we consider only the transfer operators for standard coarsening in the vertex- and cell-centered cases.

Restriction for vertex-centered grids

The most obvious restriction operator is the so-called *injection*, which is defined in two dimensions as follows:

$$v_H = I_h^H v_h \tag{2.90}$$

where

$$(v_H)_{i,j} = (v_h)_{2i,2j}, \tag{2.91}$$

i.e. the coarse grid function v_H at a grid point (i,j) takes its value directly from the corresponding fine grid value. We note that the stencil notation is given by

$$I_h^H = [1]_h^H .$$

An alternative restriction operator is the so-called *full weighting* operator, which is defined by

$$v_H = I_h^H v_h \tag{2.92}$$

where

$$\begin{aligned}(v_H)_{i,j} = \frac{1}{16}\big[&(v_h)_{2i-1,2j-1} + (v_h)_{2i-1,2j+1} + (v_h)_{2i+1,2j-1} + (v_h)_{2i+1,2j+1} \\ &2\left((v_h)_{2i,2j-1} + (v_h)_{2i,2j+1} + (v_h)_{2i-1,2j} + (v_h)_{2i+1,2j}\right) + 4(v_h)_{2i,2j}\big],\end{aligned} \tag{2.93}$$

and the stencil is defined by

$$I_h^H = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}_h^H ,$$

i.e. the value of the coarse grid function v_H at a grid point (i,j) is a nine point weighted average of the value of the fine grid function at that point and the eight points surrounding it on the fine grid. Another restriction operator is the so-called *half-weighting* operator, which is a five point weighted average, defined in two dimensions by

$$v_H = I_h^H v_h \tag{2.94}$$

where

$$(v_H)_{i,j} = \frac{1}{8}\left[(v_h)_{2i,2j-1} + (v_h)_{2i,2j+1} + (v_h)_{2i-1,2j} + (v_h)_{2i+1,2j+1} + 4(v_h)_{2i,2j}\right] \tag{2.95}$$

and the stencil is

$$I_h^H = \frac{1}{8}\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}_h^H .$$

Interpolation for vertex-centered grids

The most commonly used interpolation operator is the so-called *bilinear interpolation*, which is defined by

$$v_h = I_H^h v_H \tag{2.96}$$

where

$$
\begin{aligned}
(v_h)_{2i,2j} &= (v_H)_{i,j} \\
(v_h)_{2i+1,2j} &= \frac{1}{2}\left((v_H)_{i,j} + (v_H)_{i+1,j}\right) \\
(v_h)_{2i,2j+1} &= \frac{1}{2}\left((v_H)_{i,j} + (v_H)_{i,j+1}\right) \\
(v_h)_{2i+1,2j+1} &= \frac{1}{4}\left((v_H)_{i,j} + (v_H)_{i+1,j} + (v_H)_{i,j+1} + (v_H)_{i+1,j+1}\right)
\end{aligned} \tag{2.97}
$$

for $0 \leq i \leq n/2 - 1$ and $0 \leq i \leq m/2 - 1$, which can be represented by the stencil

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

This means that for fine grid points which are also coarse grid points the value of the fine grid function is transferred directly from the coarse grid value. For fine grid points on a horizontal coarse grid line but not a vertical one the fine grid value is the average of the values at the 2 coarse grid points. Similarly, we can use the analogous result for fine grid points on a vertical coarse grid line but not a horizontal one. For fine grid points in the middle of four coarse grid points the fine grid value is the average of the coarse grid values at the 4 points.

Restriction for cell-centered grids

For a cell-centered discretisation, each cell of the coarse grid Ω^H contains within it 4 fine grid cells and each grid point of Ω^H is surrounded by 4 grid points of Ω^h. The four cell average restriction operator evaluates the value of a coarse grid function v_H at a coarse grid point by taking the average value of the fine grid function v_h at four fine grid points surrounding it. This restriction operator can be defined formally by

$$
v_H = I_h^H v_h \tag{2.98}
$$

where

$$
(v_H)_{i,j} = \frac{1}{4}\left((v_h)_{2i-1,2j-1} + (v_h)_{2i-1,2j} + (v_h)_{2i,2j-1} + (v_h)_{2i,2j}\right), \tag{2.99}
$$

which can be given by the stencil

$$
\frac{1}{4}\begin{bmatrix} 1 & & 1 \\ & \cdot & \\ 1 & & 1 \end{bmatrix}_h^H .
$$

Interpolation for cell-centered grids

The simplest cell-centered interpolation operator simply transfers the value at a coarse grid point directly to the four grid points contained within that coarse grid cell, i.e.

$$
v_h = I_H^h v_H \tag{2.100}
$$

where

$$
(v_h)_{2i,2j} = (v_h)_{2i,2j-1} = (v_h)_{2i-1,2j} = (v_h)_{2i-1,2j-1} = (v_H)_{i,j} \tag{2.101}
$$

for $i = 1, \ldots, n/2$ and $i = 1, \ldots, m/2$. The stencil is

$$\begin{bmatrix} 1 & & 1 \\ & \cdot & \\ 1 & & 1 \end{bmatrix}_H^h .$$

The cell-centered bilinear interpolation operator is defined in a similar way as discussed in the case of the vertex-centered discretisation as follows:

$$v_h = I_H^h v_H \tag{2.102}$$

where

$$\begin{aligned}
(v_h)_{2i,2j} &= \frac{1}{16}\left[9(v_H)_{i,j} + 3\left((v_H)_{i+1,j} + (v_H)_{i,j+1}\right) + (v_H)_{i+1,j+1}\right] \\
(v_h)_{2i+1,2j} &= \frac{1}{16}\left[9(v_H)_{i+1,j} + 3\left((v_H)_{i,j} + (v_H)_{i+1,j+1}\right) + (v_H)_{i,j+1}\right] \\
(v_h)_{2i,2j+1} &= \frac{1}{16}\left[9(v_H)_{i,j+1} + 3\left((v_H)_{i,j} + (v_H)_{i,j+1}\right) + (v_H)_{i+1,j}\right] \\
(v_h)_{2i+1,2j+1} &= \frac{1}{16}\left[9(v_H)_{i+1,j+1} + 3\left((v_H)_{i+1,j} + (v_H)_{i,j+1}\right) + (v_H)_{i,j}\right]
\end{aligned} \tag{2.103}$$

for $i = 1, \ldots, n/2 - 1$ and $i = 1, \ldots, m/2 - 1$, which can be represented by the stencil

$$\frac{1}{16}\begin{bmatrix} 1 & 3 & 3 & 1 \\ 3 & 9 & 9 & 3 \\ 3 & 9 & 9 & 3 \\ 1 & 3 & 3 & 1 \end{bmatrix}_H^h .$$

Order of interpolation and restriction

An interpolation operator is said to have order $k + 1$ if it can transfer exactly polynomials of order k, 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. For example bilinear interpolation in both the vertex and cell-centered cased has order 2.

When constructing MG methods the summation of the order of the restriction and interpolation operators should be, as a general rule, greater than the order of the PDE being considered.

2.7.4 Local Fourier analysis (LFA)

The main idea of the smoothing analysis is to estimate how fast the high frequency components of the error of the solution in the Fourier modes are eliminated by a given smoother (an iterative relaxation method used to smooth the error). As is well known the so-called *local Fourier analysis* (LFA) is a power tool for analysing the MG methods, in particular the smoothing effect of any smoother.

In two-dimensional cases, the LFA studies the actions of linear operators with constant coefficients (i.e. operators L_h whose stencil entries $L_{p.q}$ are not dependant on position p in the

grid) on the grid functions characterized by

$$\varphi_{\boldsymbol{h}}(\boldsymbol{\theta},\mathbf{x}) = e^{\mathbf{i}\boldsymbol{\theta}\mathbf{x}/\boldsymbol{h}} = e^{\mathbf{i}\theta_1 x_i/h} e^{\mathbf{i}\theta_2 y_j/k} \ (\mathbf{i}=\sqrt{-1}) \tag{2.104}$$

over an infinite grid

$$\Omega_h^{\infty} = \left\{\mathbf{x} = (x_i, y_j) = (ih, jk) | (i,j) \in \mathbb{Z}^2\right\}$$

with grid spacing $\boldsymbol{h} = (h,k) = (1/n, 1/m)$ for a vertex-centered discretisation (for a cell-centered discretisation the grid points (x_i, y_j) are in different positions).

Assuming that the frequency $\boldsymbol{\theta} = (\theta_1, \theta_2)$ varies continuously in $\mathbb{R}^2$, it is not difficult to see that

$$\varphi_{\boldsymbol{h}}(\boldsymbol{\theta},\mathbf{x}) = \varphi_{\boldsymbol{h}}(\boldsymbol{\theta}',\mathbf{x}) \text{ for } \mathbf{x} \in \Omega_h^{\infty} \tag{2.105}$$

when the differences between θ_1 and θ_1' and θ_2 and θ_2' are multiple of 2π. Due to the periodic nature of the grid functions $\varphi_{\boldsymbol{h}}(\boldsymbol{\theta},\mathbf{x})$, it is enough to consider $\varphi_{\boldsymbol{h}}(\boldsymbol{\theta},\mathbf{x})$ for all $\boldsymbol{\theta} \in [-\pi,\pi)^2 = \boldsymbol{\Theta}$ [134]. With respect to standard coarsening, *low frequency components* are $\varphi_{\boldsymbol{h}}(\boldsymbol{\theta},\mathbf{x})$ such that $\boldsymbol{\theta} = (\theta_1,\theta_2) \in \boldsymbol{\Theta}_{\text{low}} = [-\pi/2, \pi/2)^2$ and *high frequency components* are $\varphi_{\boldsymbol{h}}(\boldsymbol{\theta},\mathbf{x})$ such that $\boldsymbol{\theta} = (\theta_1,\theta_2) \in \boldsymbol{\Theta}_{\text{high}} = [-\pi,\pi)^2 \backslash [-\pi/2, \pi/2)^2$. The following theorem forms a basis for most of results in the LFA.

Theorem 2.7.1 *For $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, all grid functions $\varphi_{\boldsymbol{h}}(\boldsymbol{\theta},\mathbf{x})$ are eigenfunctions of any discrete linear operator L_h with constant coefficients and the relation*

$$L_h \varphi_{\boldsymbol{h}}(\boldsymbol{\theta},\mathbf{x}) = \widetilde{L}_h(\boldsymbol{\theta}) \varphi_{\boldsymbol{h}}(\boldsymbol{\theta},\mathbf{x}) \text{ with } \mathbf{x} \in \Omega_h^{\infty} \tag{2.106}$$

holds with

$$\widetilde{L}_h(\boldsymbol{\theta}) = \sum_{q \in \mathbb{Z}^2} L_q e^{\mathbf{i}\boldsymbol{\theta} q}. \tag{2.107}$$

Proof. The proof of this theorem can be found in [134]. ■

With the result of Theorem 2.7.1 it is straight forward to analyse the smoothing properties of a given smoother used to solve a discrete PDE $L_h u_h = f_h$. Here we need to assume that one step of this smoother can be written locally as

$$L_h^+ v_h^{new} + L_h^- v_h^{old} = f_h, \tag{2.108}$$

where v_h^{old} and v_h^{new} are respectively the approximation solutions before and after applying the smoother step and

$$L_h = L_h^+ + L_h^-. \tag{2.109}$$

Subtracting (2.108) from the original discrete system $L_h u_h = f_h$ leads to

$$L_h^+ e_h^{new} + L_h^- e_h^{old} = 0, \tag{2.110}$$

which is equivalent to

$$e_h^{new} = -\frac{L_h^-}{L_h^+} e_h^{old} = S_h e_h^{old}. \tag{2.111}$$

Recall that $e_h^{old} = v_h^{old} - u_h$ and $e_h^{new} = v_h^{new} - u_h$ are the errors of the approximation solutions before and after applying the smoother step, respectively.

From (2.110) and Theorem 2.7.1 we can see that all grid functions $\varphi_{\boldsymbol{h}}(\boldsymbol{\theta}, \mathbf{x})$ are eigenfunctions of $S_h = -L_h^- / L_h^+$ and

$$S_h \varphi_{\boldsymbol{h}}(\boldsymbol{\theta}, \mathbf{x}) = \widetilde{S}_h(\boldsymbol{\theta}) \varphi_{\boldsymbol{h}}(\boldsymbol{\theta}, \mathbf{x}) = -\frac{\widetilde{L}_h^-(\boldsymbol{\theta})}{\widetilde{L}_h^+(\boldsymbol{\theta})} \varphi_{\boldsymbol{h}}(\boldsymbol{\theta}, \mathbf{x}) \text{ for } L_h^+(\boldsymbol{\theta}) \neq 0. \tag{2.112}$$

The *local smoothing factor* μ_{loc} of a given smoother is therefore defined by

$$\mu_{\text{loc}} = \sup\{|\widetilde{S}_h(\boldsymbol{\theta})| \mid \boldsymbol{\theta} \in \boldsymbol{\Theta}_{\text{high}}\}. \tag{2.113}$$

For a smoother to be effective, we hope $\mu_{\text{loc}} < 1$ and practically $\mu_{\text{loc}} = 0.75$ for instance.

Example 2.7.1 *Let us consider the GS-LEX method applied to the discrete Poisson's equation as represented by* (2.48). *We can see that*

$$L_h^+ = \frac{1}{h^2} \begin{bmatrix} 0 & 0 & 0 \\ -1 & 4 & 0 \\ 0 & -1 & 0 \end{bmatrix} \text{ and } L_h^- = \frac{1}{h^2} \begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix}.$$

Therefore the local smoothing factor for this relaxation method is given by

$$\mu_{loc} = \sup\{|\widetilde{S}_h(\boldsymbol{\theta})| \mid \boldsymbol{\theta} \in \boldsymbol{\Theta}_{high}\} \tag{2.114}$$

where $\widetilde{L}_h^+(\boldsymbol{\theta}) = \frac{1}{h^2}\left(4 - e^{-\mathrm{i}\theta_1} - e^{-\mathrm{i}\theta_2}\right)$, $\widetilde{L}_h^-(\boldsymbol{\theta}) = \frac{1}{h^2}\left(-e^{\mathrm{i}\theta_1} - e^{\mathrm{i}\theta_2}\right)$ *and*

$$\widetilde{S}_h(\boldsymbol{\theta}) = -\frac{\widetilde{L}_h^-(\boldsymbol{\theta})}{\widetilde{L}_h^+(\boldsymbol{\theta})} = \frac{e^{\mathrm{i}\theta_1} + e^{\mathrm{i}\theta_2}}{4 - e^{-\mathrm{i}\theta_1} - e^{-\mathrm{i}\theta_2}}.$$

It is shown in [134] that the supremum of (2.114) *is attained precisely at* $\boldsymbol{\theta} = (\theta_1, \theta_2) = (\pi/2, \cos^{-1}(4/5))$ *leading to* $\mu_{loc} = 0.5$.

Remark 2.7.2

1. *Although the Jacobi and GS-LEX methods including their line analogues can be written in the form* (2.108), *the GS-RB method cannot. However, LFA can still be used to analyse GS-RB type smoothers but the analysis is more involved; see [134, 140] for more details.*

2. *As noted by [134] any general discrete operator, nonlinear with nonconstant coefficients, can be linearised locally and can be replaced locally (by freezing the coefficients) by an operator with constant coefficients. In other word, LFA is still a very useful tool for analysing MG methods for nonlinear problems.*

2.7.5 Multigrid cycles

In §2.7.1 we explained the MG principles and introduced the two–grid correction scheme. Each coarse-gird correction step requires the residual equation to be solved exactly on the coarse grid Ω^H. Although $\Omega^H = \Omega^{2h}$ has 4 times fewer grid points than Ω^h for standard coarsening, a direct solver for the coarse-grid problem is still likely to be prohibitively expensive when the

discrete system is large. We could use a uni-grid iterative relaxation method, e.g. a given smoother. However, a better approach might be to use coarse-grid correction again, i.e. solve the residual equation on Ω^{2h} by relaxing on its residual equation on the next coarser grid Ω^{4h} (a grid whose grid spacing is twice that of Ω^{2h}). This process can be used recursively to solve the residual equations until we reach some very coarse grid Ω^{ph} in such a way that the corresponding residual equation can be solved exactly using a direct method at a very low computational cost. If on each coarse grid μ coarse-grid correction steps are used to approximately solve the residual equation we have what is known as a $\mu-$cycle MG step. A $\mu-$cycle MG step to update the approximation to a linear system $L_h u_h = f_h$ on the finest grid Ω^h is denoted by

$$[v_h] \leftarrow MGCYC(v_h, f_h, L_h, \nu_1, \nu_2, \mu)$$

where *Smoother* represents the results from one step of a given smoother and may be summarised in Algorithm 2.7.1. For practical applications only $\mu = 1$ or 2 is used. These methods are known as the MG V- and W-cycle, respectively. The diagrams of grids for a 4-grid MG V- and W-cycle are shown in Figure 2.5.

Algorithm 2.7.1 (MG cycle)

Denote MG parameters as follows:

ν_1 pre-smoothing steps on each level

ν_2 post-smoothing steps on each level

μ the number of multigrid cycles on each level ($l = 1$ for V$-$cycling and $l = 2$ for W$-$cycling).

$$[v_h] \leftarrow MGCYC(v_h, f_h, L_h, \nu_1, \nu_2, \mu)$$

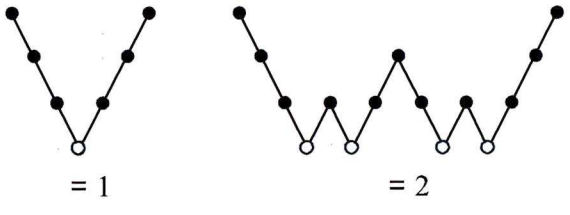


Figure 2.5: Left-Right: Illustration of grids for a 4-grid MG V-cycle ($\mu = 1$) and MG W-cycle ($\mu = 2$).

- If $\Omega^h = coarset\ grid$, solve $L_h u_h = f_h$ using a direct solver and then stop.
  Else continue with following step.
- Pre-smoothing:
  For $k = 1$ to ν_1, $[v_h] \leftarrow Smoother(v_h, f_h, L_h)$
- Restriction to the coarse grid:
  $v_H \leftarrow I_h^H v_h$
- Set the initial solution for the coarse-grid problem:
  $v_H \leftarrow 0$
- Compute the new right-hand side for the coarse-grid problem:
  $f_H \leftarrow I_h^H (f_h - L_h v_h)$
- Implement the $\mu-$cycle MG step on the coarse-grid problem:
  For $k = 1$ to μ, $[v_H] \leftarrow MGCYC\,(v_H, f_H, L_H, \nu_1, \nu_2, \mu)$
- Add the coarse-grid corrections:
  $v_h \leftarrow v_h + I_H^h v_H$
- Post-smoothing:
  For $k = 1$ to ν_2, $[v_h] \leftarrow Smoother(v_h, f_h, L_h)$

2.7.6 Full multigrid methods

A full multigrid (FMG) method is designed to provide reliable initial solutions for iterative solvers including MG methods. The idea is to solve first the original problem at the coarsest grid and then interpolate the obtained solution as the excellent initial solution to the next finer grid problem level by level until it reaches the finest grid. This can be shown in Figure 2.6 for a 4-grid problem and summarised as follows:

Algorithm 2.7.2 (FMG method)

Denote FMG parameters as follows:

ν_1 pre-smoothing steps on each level

ν_2 post-smoothing steps on each level

μ the number of multigrid cycles on each level ($l = 1$ for V$-$cycling and $l = 2$ for W$-$cycling).

$$[v_h] \leftarrow FMG(v_h, f_h, L_h, \nu_1, \nu_2, \mu)$$

- If $\Omega^h = coarset\ grid$, solve $L_h u_h = f_h$ using a direct solver and then stop.
  Else continue with following step.
- Restriction to the coarse grid:
  $v_H \leftarrow I_h^H v_h, \qquad f_H \leftarrow I_h^H f_h$
- Implement the FMG step on the next coarser grid:
  $[v_H] \leftarrow FMG\,(v_H, f_H, L_H, \nu_1, \nu_2, \mu)$
- Interpolation to the next finer grid:
  $v_h \leftarrow I_H^h v_H$
- Implement the MG $\mu-$cycle on the next finer grid :
  $[v_H] \leftarrow MGCYC\,(v_h, f_h, L_h, \nu_1, \nu_2, \mu)$

2.7.7 Computational work

To estimate a computational work of a V- (or W-) cycle MG method it is usually expressed in terms of work units (WUs). Here we define a WU as the computational cost of performing

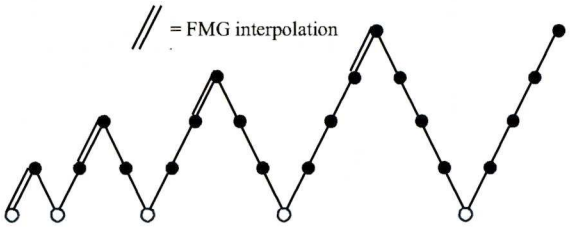


Figure 2.6: The scheme illustrates the typical structure of a FMG method.

a smoother or relaxation step on the finest grid. In general the cost of transfer operators can be neglected since it is no more than 20% of the cost of the entire cycle. A V-cycle for a d-dimensional problem with $\nu_1 = \nu_2 = 1$, where ν_1 and ν_2 are the numbers of pre- and post-smoothing steps, requires p^{-d} WUs per each grid Ω^{pd}. Thus the total costs of one V-cycle used n coarse grids can be estimated by

$$\text{V-cycle costs} = 2\left\{1 + 2^{-d} + 2^{-2d} + \ldots + 2^{-nd}\right\} < \frac{2}{1 - 2^{-nd}} \text{ WUs.}$$

For instance, a single V-cycle has a cost of about $\frac{8}{3}$ WUs for a 2-dimensional problem.

2.7.8 Full approximation scheme nonlinear multigrid method

Full approximation scheme nonlinear multigrid (FAS-NMG) method has become an efficient approach for solving nonlinear problems. Here instead of a discrete linear PDE we have a discrete nonlinear PDE,

$$N_h u_h = f_h \tag{2.115}$$

involving the nonlinear operator N_h acting on u_h. Let v_h be the result computed by performing a few steps with a nonlinear smoother like the ones discussed in §2.6.7 on the fine grid Ω_h. Therefore the nonlinear residual equation on the fine grid Ω_h is given by

$$N_h u_h - N_h v_h = N_h(v_h + e_h) - N_h v_h = f_h - N_h v_h = r_h, \tag{2.116}$$

where $e_h = u_h - v_h$ is the error of the solution and $r_h = f_h - N_h v_h$ is the nonlinear residual. In order to correct the approximated solution v_h on Ω_h, one needs to compute the error e_h. However, the error cannot be computed directly on Ω_h. We then need to transfer the following nonlinear residual equation to the coarse grid Ω_H as follows

$$\underbrace{N_h(v_h + e_h)}_{N_h u_h} = \underbrace{r_h + N_h v_h}_{f_h} \quad \rightarrow \quad \underbrace{N_H(v_H + e_H)}_{N_H u_H} = \underbrace{r_H + N_H v_H}_{f_H} \tag{2.117}$$

After the nonlinear residual equation (2.117) on the coarse grid have been solved with a method of our choice, the coarse-grid correction $e_H = u_H - v_H$ is then interpolated back to the fine grid e_h that can now be used for updating the approximated solution v_h of the original nonlinear system on the fine grid $v_h^{new} = v_h + e_h$. As discussed for the linear case in §2.7.1 the last step

is to perform the nonlinear smoother again to remove high frequency parts of the interpolated error.

Obviously we can extend the 2-grid FAS-NMG method represented by the above procedure to a MG method. We employ coarse-grid correction recursively to solve the nonlinear residual equation until we reach to some very coarse grid. We note first that we may have to solve the nonlinear residual equation using a given nonlinear smoother or another iterative method on the coarsest grid. We also note further that we use the initial guess v_H for a solution to the nonlinear residual equation on Ω_H because we are working with the full approximation scheme. This is different from the linear case where we use an initial guess 0 for the solution to the residual equation on Ω_H.

Finally, a FAS-NMG method can be summarised as represented in Algorithm 2.7.3. Recall that *Smoother* means a nonlinear smoother (i.e. a nonlinear relaxation technique) with suitable smoothing properties.

Algorithm 2.7.3 (FAS-NMG method)

Denote FAS-NMG parameters as follows:

ν_1 pre-smoothing steps on each level

ν_2 post-smoothing steps on each level

μ the number of multigrid cycles on each level ($l = 1$ for V−cycling and $l = 2$ for W−cycling).

$$[v_h] \leftarrow FASCYC(v_h, f_h, N_h, \nu_1, \nu_2, \mu)$$

- If $\Omega^h = coarset\ grid$, solve $N_h u_h = f_h$ using a given nonlinear smoother. or another iterative method. Else continue with following step.
- Pre-smoothing:
  For $k = 1$ to ν_1, $[v_h] \leftarrow Smoother(v_h, f_h, N_h)$
- Restriction to the coarse grid:
  $v_H \leftarrow I_h^H v_h, r_H = I_h^H (f_h - N_h v_h)$
- Set the initial solution for the coarse-grid problem:
  $\overline{v}_H \leftarrow v_H$
- Compute the new right-hand side for the coarse-grid problem:
  $f_H \leftarrow r_H + N_H v_H$
- Implement the μ−cycle FAS-NMG step on the coarse-grid problem:
  For $k = 1$ to μ, $[v_H] \leftarrow FASCYC\,(v_H, f_H, N_H, \nu_1, \nu_2, \mu)$
- Add the coarse-grid corrections:
  $v_h \leftarrow v_h + I_H^h (v_H - \overline{v}_H)$
- Post-smoothing:
  For $k = 1$ to ν_2, $[v_h] \leftarrow Smoother(v_h, f_h, N_h)$

Chapter 3

Variational Image Registration

This chapter presents a general framework for image registration. This framework is based on a variational formulation of the registration problem and its solution schemes to be considered are based on the so-called optimise-discretise and discretise-optimise approaches. This general concept will be specified for various registration techniques in the next chapters.

3.1 Images

Physically, an image is a set of measurements obtained by integration of some density field, e.g. radiation, over a finite area or volume. In some applications images are vector valued, as colour images for example. In this thesis, they are restricted to scalar or gray intensity images and modelled as continuous mappings from an image domain $\Omega \subset \mathbb{R}^d$ into $V \subset \mathbb{R}_0^+$, where $d \in \mathbb{N}$ represents the spatial dimension of the images which is usually $d = 2$ (images) or $d = 3$ (volume data set) with smooth boundary $\partial\Omega$. This means that an image $I : \Omega \to V$ associates with each spatial position $\mathbf{x} = (x_1, x_2, \ldots, x_d)^\top \in \Omega$ its gray intensity value $I(\mathbf{x})$.

3.2 Variational formulation of the registration problem

A general framework of the image registration can be formulated as follows: Given two images of the same object (or similar ones) which are referred to as a *reference* R and a *template* T, we search for a vector valued transformation φ defined by

$$\varphi(\boldsymbol{u})(\cdot) : \mathbb{R}^d \to \mathbb{R}^d, \qquad \varphi(\boldsymbol{u})(\mathbf{x}) : \mathbf{x} \mapsto \mathbf{x} + \boldsymbol{u}(\mathbf{x})$$

that depends on the unknown *deformation* or *displacement field*

$$\boldsymbol{u} : \mathbb{R}^d \to \mathbb{R}^d, \qquad \boldsymbol{u} : \mathbf{x} \mapsto \boldsymbol{u}(\mathbf{x}) = (u_1(\mathbf{x}), u_2(\mathbf{x}), \ldots, u_d(\mathbf{x}))^\top.$$

such that the transformed template

$$T \circ \varphi(\boldsymbol{u}(\mathbf{x})) = T(\mathbf{x} + \boldsymbol{u}(\mathbf{x})) = T_{\boldsymbol{u}}(\mathbf{x}) = T_{\boldsymbol{u}}$$

becomes similar to the reference R. Once the corresponding location $\varphi(\boldsymbol{u}(\mathbf{x})) = \mathbf{x} + \boldsymbol{u}(\mathbf{x})$ is calculated for each spatial location $\mathbf{x} \in \Omega$, an interpolation or approximation step, e.g. d-linear

interpolation, is required to assign the image intensity values for the transformed template $T_{\boldsymbol{u}}$ at non-grid locations within image boundaries; see Figure 3.1. For locations outside the image boundaries, the image intensities are usually set to be a constant value (typically zero [104]). Note that the term $\boldsymbol{u}$ is used to model the transformation φ because it can view as how a point in the transformed template $T_{\boldsymbol{u}}$ is moved away from its original position. Thus the problem of finding the transformation φ and the deformation field $\boldsymbol{u}$ that the transformed template $T_{\boldsymbol{u}}$ matches the reference R is equivalent. Here the geometric transformation φ can be alternatively defined by $\varphi(\boldsymbol{u})(\mathbf{x}) : \mathbf{x} \mapsto \mathbf{x} - \boldsymbol{u}(\mathbf{x})$; see [104].

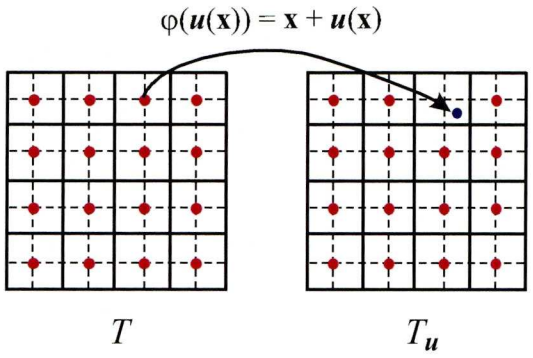


Figure 3.1: The concept of the image registration visualised as the mapping between two images T and $T_{\boldsymbol{u}}$. An interpolation scheme has to be employed to assign the image intensity values in the transformed image $T_{\boldsymbol{u}}$ if the transformed position $\varphi(\boldsymbol{u}(\mathbf{x})) = \mathbf{x} + \boldsymbol{u}(\mathbf{x})$ does not lie on the integer $\mathbf{x} = (x_1, x_2)^\top$ grid point.

All registration strategies require a suitable similarity functional (sometimes also called similarity or distance measure) $\mathcal{D}$ which measures the disparity or similarity between the transformed template $T_{\boldsymbol{u}}$ and the reference R over the image domain. Thus, the image registration problem can be formulated as the minimisation problem of $\mathcal{D}$ in the following manner:

$$\min_{\boldsymbol{u}} \mathcal{D}(\boldsymbol{u}) \tag{3.1}$$

As is known, the image registration problem (3.1) is a nonlinear and ill-posed one in the sense of Hadamard because the direct minimisation of $\mathcal{D}$ will not guarantee a unique solution for $\boldsymbol{u}$. It becomes necessary to impose a *deformation model* $\mathcal{R}$, which is also known as a *regularising constraint* or *regulariser*, on the solution $\boldsymbol{u}$ for penalising unwanted and irregular solutions using priori knowledge. This approach is mathematically known as regularisation. As a consequence, the image registration problem can be posed as a minimisation problem of the *joint* functional:

$$\min_{\boldsymbol{u}} \{\mathcal{J}_\alpha(\boldsymbol{u}) = \mathcal{D}(\boldsymbol{u}) + \alpha\mathcal{R}(\boldsymbol{u})\}. \tag{3.2}$$

Here $\alpha > 0$ is the regularisation parameter that compromises similarity and regularity, and $\boldsymbol{u}$ is searched over a set $\mathcal{U}$ of admissible functions minimising $\mathcal{J}_\alpha$. The set $\mathcal{U}$ is generally assumed to be a Hilbert space $\mathcal{H}$ equipped with its usual scalar product

$$\langle \boldsymbol{u}, \boldsymbol{\eta} \rangle_{\mathcal{H}} = \int_\Omega \boldsymbol{u}(\mathbf{x}) \cdot \boldsymbol{\eta}(\mathbf{x})\, d\mathbf{x} = \int_\Omega \langle \boldsymbol{u}(\mathbf{x}), \boldsymbol{\eta}(\mathbf{x}) \rangle_{\mathbb{R}^d}\, d\mathbf{x}.$$

Recall that $\langle \cdot, \cdot \rangle_{\mathbb{R}^d}$ denotes the Euclidean scalar product.

In general we expect minimisers of the energy functional $\mathcal{J}_\alpha$ to exist. A necessary condition for a (local) minimiser $\boldsymbol{u}$ of $\mathcal{J}_\alpha$ is that the Gâteaux derivative (or the first variation) $\delta\mathcal{J}_\alpha(\boldsymbol{u};\boldsymbol{\eta})$ of $\mathcal{J}_\alpha$ must vanish for all variational directions $\boldsymbol{\eta} \in \mathcal{H}$, i.e.

$$\delta\mathcal{J}_\alpha(\boldsymbol{u};\boldsymbol{\eta}) = \frac{d}{d\epsilon}\,\mathcal{J}_\alpha(\boldsymbol{u}+\epsilon\boldsymbol{\eta})|_{\epsilon=0} = \lim_{\epsilon\to 0}\frac{\mathcal{J}_\alpha(\boldsymbol{u}+\epsilon\boldsymbol{\eta}) - \mathcal{J}_\alpha(\boldsymbol{\eta})}{\epsilon} = 0. \tag{3.3}$$

Note that if $\delta\mathcal{J}_\alpha(\boldsymbol{u};\boldsymbol{\eta}) = \langle \nabla_{\boldsymbol{u}}\mathcal{J}_\alpha, \boldsymbol{\eta}\rangle_{\mathcal{H}}$, $\nabla_{\boldsymbol{u}}\mathcal{J}_\alpha$ defines the gradient of the joint functional $\mathcal{J}_\alpha$. Then, the necessary condition of optimality is readily equivalent to $\nabla_{\boldsymbol{u}}\mathcal{J}_\alpha = 0$, which is known as the Euler-Lagrange equation associated with the minimisation problem (3.2) and can be easily computed when $\mathcal{J}_\alpha$ is represented in a simple form. For registration purposes, the joint functional $\mathcal{J}_\alpha$ is generally defined by

$$\mathcal{J}_\alpha(\boldsymbol{u}) = \int_\Omega F(\mathbf{x}, \boldsymbol{u}(\mathbf{x}), \nabla\boldsymbol{u}(\mathbf{x}))d\mathbf{x} \tag{3.4}$$

with a functional F assumed to have continuous partial derivatives with respect to each of its arguments. One can show by extending the results from Lemma 2.2.1 to the vectorial case that its Euler-Lagrange PDEs becomes

$$\nabla_{\boldsymbol{u}}F - \nabla\cdot\nabla_{\nabla\boldsymbol{u}}F = 0 \text{ on } \Omega. \tag{3.5}$$

Here we denote by $\nabla_{\boldsymbol{u}}F = (\partial F/\partial u_1, \ldots, \partial F/\partial u_d)^\top$ the gradient of F with respect to its second argument $\boldsymbol{u}(\mathbf{x})$ and in a similar way the gradient of F with respect to $\nabla\boldsymbol{u}(\mathbf{x})$, i.e. its third argument, is given by

$$\nabla_{\nabla u}F = \begin{bmatrix} \partial F/\partial u_{1,1} & \cdots & \partial F/\partial u_{1,d} \\ \vdots & & \vdots \\ \partial F/\partial u_{d,1} & \cdots & \partial F/\partial u_{d,d} \end{bmatrix} \in \mathbb{R}^{d\times d},$$

where $u_{i,j}$ is an abbreviation for $\partial u_i/\partial x_j$. As in the scalar case represented in §2.2, the boundary conditions of (3.5) are essential conditions when they are imposed explicitly. If, in contrast, boundary conditions are not given explicitly, we are dealing with natural conditions corresponding to the minimisation problem (3.2):

$$\langle \nabla_{\nabla\boldsymbol{u}_l}F, \boldsymbol{n}\rangle_{\mathbb{R}^d} = 0 \text{ on } \partial\Omega,\ l = 1, ..., d. \tag{3.6}$$

Recall that $\boldsymbol{n}$ denotes the outer normal unit vector of $\partial\Omega$.

We can see that for an image registration problem the task of finding a minimiser $\boldsymbol{u}$ of $\mathcal{J}_\alpha$ and the task of solving the Euler-Lagrange equation for $\boldsymbol{u}$ is equivalent. Together with boundary conditions, in this context (3.2) is called the *variational formulation* of the registration problem (3.5). Without loss of generality we assume that the registration problem is described in the two-dimensional case ($d = 2$) throughout this thesis, but it is readily extendable to the three-dimensional case ($d = 3$). We also assume further that $\Omega = [0,1]^2 \subset \mathbb{R}^2$ and $V = [0,1]$ for 2D gray intensity images.

3.3 Similarity measures

As mentioned in the previous section, a similarity measure is related to image similarity and used to provide a quantitative measure for the quality of the transformation. Several approaches have been proposed. These measures are based on either the so-called *feature-* or *intensity-based approaches*. For the first approach the calculations are based on a number of outstanding correspondences which are well-selected manually or automatically from the given images, such as landmarks[1] or a combination of curves and surfaces. It is recommended when both images contain enough distinctive and easily detectable features; see [104, p.31 and 44]. In contrast to the first approach, the latter approach is more general and robust than the first approach. The basic idea is to use the whole (full raw) information of the given images and can be defined in terms of functionals in $T_{\boldsymbol{u}}$ and R as follows:

$$\mathcal{D}(\boldsymbol{u}) = \mathcal{D}(T_{\boldsymbol{u}}, R) = \int_{\Omega} F_{\mathcal{D}}(T(\mathbf{x} + \boldsymbol{u}(\mathbf{x})), R(\mathbf{x}))\, d\mathbf{x}.$$

Due to its robustness the second approach is adopted in this thesis.

3.3.1 Sum of squared differences (SSD)

When the image intensities of the given images are comparable (i.e., in a monomodal registration scenario), the proper choice of similarity measures is the so-called *sum of squared differences* (SSD) [46, 47, 48, 49, 51, 53, 54, 62, 65, 59, 60, 61, 76, 72, 79, 78, 73, 75, 83, 89, 90, 94, 104, 131, 145] :

$$\mathcal{D}^{\mathrm{SSD}}(\boldsymbol{u}) = \frac{1}{2}\int_{\Omega} (T(\mathbf{x} + \boldsymbol{u}(\mathbf{x})) - R(\mathbf{x}))^2\, d\mathbf{x}, \tag{3.7}$$

which is adopted mainly in this thesis, and its Gâteaux derivative is given by

$$\delta\mathcal{D}^{\mathrm{SSD}}(\boldsymbol{u};\boldsymbol{v}) = \left\langle \nabla_{\boldsymbol{u}}\mathcal{D}^{\mathrm{SSD}}, \boldsymbol{\eta}\right\rangle_{\mathcal{H}}, \tag{3.8}$$

where

$$\nabla_{\boldsymbol{u}}\mathcal{D}^{\mathrm{SSD}} = (T_{\boldsymbol{u}} - R)\nabla_{\boldsymbol{u}}T_{\boldsymbol{u}}. \tag{3.9}$$

3.3.2 Mutual information (MI)

In general, if the images are recorded with different imaging devices or modalities (i.e. in a multimodal registration scenario), the $\mathcal{D}^{\mathrm{SSD}}$ functional is not appropriate. The main reason is that the same structure may have totally different intensity values. In this case, the common choice of similarity measures is the so-called *mutual information* (MI). For convenience, let $i_1 = R(\mathbf{x})$ and $i_2 = T(\mathbf{x} + \boldsymbol{u}(\mathbf{x}))$ be the intensity values of the reference and the transformed template. We suppose that i_1 and i_2 are (continuous) random variables whose probability density functions[2] are given by $p^R(i_1)$ and $p^{T_{\boldsymbol{u}}}_{\boldsymbol{u}}(i_2)$, respectively. We denote by $p^{R,T_{\boldsymbol{u}}}_{\boldsymbol{u}}(i_1, i_2)$

[1] A landmark or maker is the location of a typically outstanding feature of an image, such as the tip of a finger or the point of maximum curvature.

[2] A probability density function (abbreviated as pdf, or just density) of a continuous random variable is a function that describes the relative likelihood for this random variable to occur at a given point in the observation space.

the joint density , which summarises the co-occurrence of events from i_1 and i_2 and describes how random the joint variable (i_1, i_2) is. The MI is defined by the Kullback-Leibler distance between the joint distribution $p_{\boldsymbol{u}}^{R,T_{\boldsymbol{u}}}(i_1, i_2)$ and the product distribution $p^R(i_1) \cdot p_{\boldsymbol{u}}^{T_{\boldsymbol{u}}}(i_2)$ of the random variables i_1 and i_2 as follows:

$$\mathcal{D}^{\mathrm{MI}}(\boldsymbol{u}) = \int_{\mathbb{R}^2} p_{\boldsymbol{u}}^{R,T_{\boldsymbol{u}}}(i_1, i_2) \log\left(\frac{p_{\boldsymbol{u}}^{R,T_{\boldsymbol{u}}}(i_1, i_2)}{p^R(i_1) \cdot p_{\boldsymbol{u}}^{T_{\boldsymbol{u}}}(i_2)}\right) di_1 di_2, \tag{3.10}$$

which is non-negative. Note that $\mathcal{D}^{\mathrm{MI}}(\boldsymbol{u}) = 0$ if and only if i_1 and i_2 are independent, i.e., $p_{\boldsymbol{u}}^{R,T_{\boldsymbol{u}}}(i_1, i_2) = p^R(i_1) \cdot p_{\boldsymbol{u}}^{T_{\boldsymbol{u}}}(i_2)$. It follows directly that if they are independent, the random variable i_1 can tell us nothing about the random variable i_2. Thus the MI is a measure of similarity between the given images. This signifies that we have to maximise $\mathcal{D}^{\mathrm{MI}}(\boldsymbol{u})$ or equivalently minimise $\mathcal{D}^{-\mathrm{MI}}(\boldsymbol{u}) = -\mathcal{D}^{\mathrm{MI}}(\boldsymbol{u})$. Following the approach of [37, 80, 81] one can show that its Gâteaux derivative is given by

$$\delta\mathcal{D}^{-\mathrm{MI}}(\boldsymbol{u}; \boldsymbol{v}) = \left\langle \nabla_{\boldsymbol{u}} \mathcal{D}^{-\mathrm{MI}}, \boldsymbol{\eta} \right\rangle_{\mathcal{H}}, \tag{3.11}$$

where

$$\nabla_{\boldsymbol{u}} \mathcal{D}^{-\mathrm{MI}} = -\frac{1}{|\Omega|}[\psi \star \frac{\partial L^{\boldsymbol{u}}}{\partial i_2}](R, T_{\boldsymbol{u}}) \nabla_{\boldsymbol{u}} T_{\boldsymbol{u}} \tag{3.12}$$

with

$$\frac{\partial L^{\boldsymbol{u}}}{\partial i_2} = \frac{1}{p_{\boldsymbol{u}}^{R,T_{\boldsymbol{u}}}(i_1, i_2)} \frac{\partial p_{\boldsymbol{u}}^{R,T_{\boldsymbol{u}}}(i_1, i_2)}{\partial i_2} - \frac{1}{p_{\boldsymbol{u}}^{T_{\boldsymbol{u}}}(i_2)} \frac{\partial p_{\boldsymbol{u}}^{T_{\boldsymbol{u}}}(i_2)}{\partial i_2}. \tag{3.13}$$

Here $|\Omega|$ denotes the area of Ω and $\psi : \mathbb{R}^2 \to \mathbb{R}$ is a smooth bidimensional density kernel used to estimate the joint density of the images R and $T_{\boldsymbol{u}}$, i.e.

$$p_{\boldsymbol{u}}^{R,T_{\boldsymbol{u}}}(i_1, i_2) = \frac{1}{|\Omega|} \int_{\Omega} \psi(R(\mathbf{x}) - i_1, T_{\boldsymbol{u}}(\mathbf{x}) - i_2) d\mathbf{x}, \tag{3.14}$$

with $\star$ denoting the convolution operator:

$$[p \star q](z_1, z_2) = \int_{\mathbb{R}^2} p(z_1 - i_1, z_2 - i_2) q(i_1, i_2) di_1 di_2. \tag{3.15}$$

Estimating the joint density

In image registration, two approaches are popular to estimate the joint density. The first approach is based on histograms and the second is based on Parzen-window estimators. Histogram-based estimators are commonly used in registration. However, it is known to have inferior approximation properties because the histogram is generally based on rounding and thus leads to nondifferential function which is not suitable for optimisation purposes [105, 127]. Nevertheless, there are some recent works developed to overcome these difficulties; see [95] for example. This section focuses only on the latter approach.

A basic idea of a Parzen-window estimator is to work with a smooth kernel function which basically spreads out sampled data. The Parzen-window estimator has better approximation properties and can give a smooth estimator which is much better suited for optimisation purposes involving partial derivatives. Let $\sigma > 0$ be the width of the Parzen-window kernel. The

Parzen-window estimator for a probability density function $p(x)$ given m samples X_1, X_2, ..., X_n is defined by

$$\widetilde{p}_\sigma(x) = \frac{1}{n}\sum_{i=1}^{n} K_\sigma(x - X_i), \tag{3.16}$$

with K being a symmetric kernel function such that $\int K(u)du = 1$ and $K_\sigma(u) = (1/\sigma)K(u/\sigma)$.

For our estimator we follow the approach of [37, 38, 71] by applying the Gaussian kernel $K_\sigma(u) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{u^2}{2\sigma^2}\right)$ where the optimal value of σ is determined by maximising the pseudo-likelihood[3] $P(\sigma)$:

$$P(\sigma) = \prod_{i=1}^{m} \widetilde{p}_\sigma(X_i). \tag{3.17}$$

Since this pseudo-likelihood has a trivial maximum at $\sigma = 0$, it has been suggested to use leave-one-out cross validation by replacing $\widetilde{p}_\sigma$ in (3.17) by

$$\widetilde{p}_{\sigma,i} = \frac{1}{n-1} \sum_{j=1, j\neq i}^{n} K_\sigma(x - X_i) \tag{3.18}$$

leading to minimise the Kullback-Leibler distance between $\widetilde{p}_\sigma(x)$ and $p(x)$. For registration purposes, the cross validation scheme is applied twice to determine σ^R and σ^{T_u} for the reference and the transformed template, respectively. We select $\sigma = \max\{\sigma^R, \sigma^{T_u}\}$ to define a 2D Parzen-window kernel $\psi_\sigma(i_1, i_2) = K_\sigma(i_1)K_\sigma(i_2)$ and the joint image intensity probability $p_{\boldsymbol{u}}^{R,T_u}$ is estimated by

$$\widetilde{p}_{\boldsymbol{u}}^{R,T_u}(i_1, i_2) = \frac{1}{m}\sum_{i=1}^{m} K_\sigma(R(\mathbf{X}_i) - i_1)K_\sigma(T_{\boldsymbol{u}}(\mathbf{X}_i) - i_2) \tag{3.19}$$

where $\mathbf{X} = \{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_m\}$ denote the sample of size m selected from grid points resulting from the discretisation of Ω. We note that the marginal densities $p^R(i_1)$ and $p_{\boldsymbol{u}}^{T_u}(i_2)$ are obtained by integrating $p_{\boldsymbol{u}}^{R,T_u}$ over rows and columns, respectively, i.e.

$$p^R(i_1) = \int_{\mathbb{R}} p_{\boldsymbol{u}}^{R,T_u}(i_1, i_2)di_2 \text{ and } p_{\boldsymbol{u}}^{T_u}(i_2) = \int_{\mathbb{R}} p_{\boldsymbol{u}}^{R,T_u}(i_1, i_2)di_1. \tag{3.20}$$

3.4 Regularisations

As is well-known, the actual choice of $\mathcal{R}$ can considerably affect on the deformation field (the solution) and the registered image. Unfortunately, choosing an appropriate $\mathcal{R}$ that fits all applications is generally difficult. Thus, care has to be taken for each application area, as image registration is inherently ill-posed as already pointed out in §3.2.

In this section the variational models with well-known $\mathcal{R}$, which have been proven to be very useful and commonly used in many registration applications, are briefly reviewed.

[3]Pseudo-likelihood is a measure in statistics that serves as an approximation of the distribution of a random variable.

3.4.1 Elastic regularisation

The elastic regularisation is the most popular choice, which is based on the linearised elastic potential of $\boldsymbol{u}$ and given by

$$\mathcal{R}^{\text{elas}}(\boldsymbol{u}) = \int_{\Omega} ((\mu/4) \sum_{l,m=1}^{2} (\partial_{x_l} u_m + \partial_{x_m} u_l)^2 + (\lambda/2)(\nabla \cdot \boldsymbol{u})^2) d\mathbf{x}, \tag{3.21}$$

where its Gâteaux derivative is given by

$$\delta\mathcal{R}^{\text{elas}}(\boldsymbol{u};\boldsymbol{v}) = \left\langle \nabla_{\boldsymbol{u}} \mathcal{R}^{\text{elas}}, \boldsymbol{\eta} \right\rangle_{\mathcal{H}}, \tag{3.22}$$

with

$$\begin{aligned} \nabla_{\boldsymbol{u}} \mathcal{R}^{\text{elas}} &= -\mu \Delta \boldsymbol{u} - (\lambda + \mu) \nabla \cdot \nabla \boldsymbol{u} \\ &= \begin{pmatrix} -((\lambda + 2\mu)\partial_{x_1 x_1} u_1 + \mu \partial_{x_2 x_2} u_1 + (\lambda + \mu)\partial_{x_1 x_2} u_2) \\ -((\lambda + \mu)\partial_{x_1 x_2} u_1 + \mu \partial_{x_1 x_1} u_2 + (\lambda + 2\mu)\partial_{x_2 x_2} u_2) \end{pmatrix} \end{aligned} \tag{3.23}$$

subject to the boundary condition $\left\langle \mu \left(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^{\top} \right) + \lambda \text{diag}(\nabla \cdot \boldsymbol{u}), \boldsymbol{n} \right\rangle_{\mathbb{R}^2} = 0$ on $\partial\Omega$. Here $\mu > 0$ and $\lambda \geq 0$ are the so-called Lamé constants which reflect material properties. This variational model, of course, allows only elastic deformations, and penalises others, in particular affine linear ones; see more details in [8, 15, 104] and references therein.

3.4.2 Diffusive regularisation

The diffusive regularisation introduced by Fischer and Modersitzki [46] is the simplest choice of $\mathcal{R}$, which is based on the L^2 norm of ∇u_l and given by,

$$\mathcal{R}^{\text{diff}}(\boldsymbol{u}) = \frac{1}{2} \sum_{l=1}^{2} \int_{\Omega} |\nabla u_l|^2 \, d\mathbf{x}. \tag{3.24}$$

For this variational model one can show that its Gâteaux derivative is given by

$$\delta\mathcal{R}^{\text{diff}}(\boldsymbol{u};\boldsymbol{v}) = \left\langle \nabla_{\boldsymbol{u}} \mathcal{R}^{\text{diff}}, \boldsymbol{\eta} \right\rangle_{\mathcal{H}}, \tag{3.25}$$

with

$$\nabla_{\boldsymbol{u}} \mathcal{R}^{\text{diff}} = -\Delta \boldsymbol{u} = \begin{pmatrix} -\Delta u_1 \\ -\Delta u_2 \end{pmatrix} \tag{3.26}$$

subject to the Neumann boundary condition $\langle \nabla u_l, \boldsymbol{n} \rangle_{\mathbb{R}^2} = 0$ on $\partial\Omega$. We note that this regularisation technique can be viewed as a typical case of the elastic regularisation when non-physical parameters, $\mu = 1$ and $\lambda = -1$, are applied. Also, it is well-known as the classical method of Horn and Schunck [84] for optical flow computation in order to smooth the deformation field $\boldsymbol{u}$.

3.4.3 Fischer and Modersitzki's curvature model based regularisation

For registration purposes, the curvature-type regularisation was introduced originally by Fischer and Modersitzki [47, 48]. Their variational model is based on an approximation of the mean curvature of the surface of u_l given by

$$\mathcal{R}^{\text{FMcurv}}(\boldsymbol{u}) = \frac{1}{2} \sum_{l=1}^{2} \int_{\Omega} (\widehat{\kappa}_M(u_l))^2 d\mathbf{x} = \frac{1}{2} \sum_{l=1}^{2} \int_{\Omega} (\Delta u_l)^2 d\mathbf{x} \tag{3.27}$$

where its Gâteaux derivative is given by

$$\delta\mathcal{R}^{\mathrm{FMcurv}}(\boldsymbol{u};\boldsymbol{v}) = \left\langle \nabla_{\boldsymbol{u}}\mathcal{R}^{\mathrm{FMcurv}}, \boldsymbol{\eta} \right\rangle_{\mathcal{H}}, \tag{3.28}$$

with

$$\nabla_{\boldsymbol{u}}\mathcal{R}^{\mathrm{FMcurv}} = \Delta^2\boldsymbol{u} = \begin{pmatrix} \Delta^2 u_1 \\ \Delta^2 u_2 \end{pmatrix} \tag{3.29}$$

subject to the typical boundary conditions $\nabla u_l = \nabla\Delta u_l = 0$ on $\partial\Omega$. Due to the second-order derivatives represented in the energy functional (3.27), this variational model leads to smoother deformation fields than those of (3.21) and (3.24) based on the first derivatives. Moreover, it does not require an additional affine linear pre-registration step to be successful. Here u_l is understood as a surface represented by $(x_1, x_2, u_l(x_1, x_2))$ where initially $u_l(x_1, x_2) = z$ and then the mean curvature of the surface of u_l is given by

$$\kappa_M(u_l) = \nabla \cdot \frac{\nabla u_l}{\sqrt{1+|\nabla u_l|^2}} = \frac{(1+u_{l_{x_1}}^2)u_{l_{x_1x_1}} - 2u_{l_{x_1}}u_{l_{x_2}}u_{l_{x_1x_2}} + (1+u_{l_{x_2}}^2)u_{l_{x_2x_2}}}{(1+u_{l_{x_1}}^2+u_{l_{x_2}}^2)^{3/2}}. \tag{3.30}$$

Assuming that $\nabla u_l \approx 0$ yields $\kappa_S(u_l) \approx \widehat{\kappa}_S(u_l) = \Delta u_l$.

3.4.4 Henn and Witsh's curvature model based regularisation

Henn and Witsh [73, 78] introduced a typical curvature-based regularisation. Their variational model is based on an approximation of the sum of the squared principal curvatures $\kappa_{P_1}(u_l)$ and $\kappa_{P_2}(u_l)$ of the surface of u_l and defined by

$$\begin{aligned} \mathcal{R}^{\mathrm{HWcurv}}(\boldsymbol{u}) &= \frac{1}{2}\sum_{l=1}^{2}\int_{\Omega}((\widehat{\kappa}_M(u_l))^2 - 2\widehat{K}_G(u_l))d\mathbf{x} \\ &= \frac{1}{2}\sum_{l=1}^{2}\int_{\Omega}(\Delta u_l)^2 - 2(u_{l_{x_1x_1}}u_{l_{x_2x_2}} - u_{l_{x_1x_2}}^2)d\mathbf{x} \end{aligned} \tag{3.31}$$

and its Gâteaux derivative is given by

$$\delta\mathcal{R}^{\mathrm{HWcurv}}(\boldsymbol{u};\boldsymbol{v}) = \left\langle \nabla_{\boldsymbol{u}}\mathcal{R}^{\mathrm{HWcurv}}, \boldsymbol{\eta} \right\rangle_{\mathcal{H}}, \tag{3.32}$$

with

$$\nabla_{\boldsymbol{u}}\mathcal{R}^{\mathrm{HWcurv}} = \Delta^2\boldsymbol{u} = \begin{pmatrix} \Delta^2 u_1 \\ \Delta^2 u_2 \end{pmatrix} \tag{3.33}$$

subject to the higher-order boundary conditions $B_1(u_l) = 0$ and $B_2(u_l) = 0$ on $\partial\Omega$ where

$$B_1(u_l) = -\frac{\partial}{\partial\boldsymbol{n}}\Delta u_l - \frac{\partial}{\partial\boldsymbol{s}}\left[\frac{\partial^2 u_l}{\partial x_1\partial x_2}(n_1^2 - n_2^2) + \left(\frac{\partial^2 u_l}{\partial^2 x_2} - \frac{\partial^2 u_l}{\partial^2 x_1}\right)n_{x_1}n_{x_2}\right], \tag{3.34}$$

$$B_2(u_l) = \frac{\partial^2 u_l}{\partial\boldsymbol{n}^2}, \tag{3.35}$$

and $\boldsymbol{s}$ denotes the tangential component vertical to $\boldsymbol{n}$. We note that the kernel of the energy functional (3.31) consists only of the affine linear displacements, and consequently the energy is invariant under planar rotation and translation. We also note that assuming $\nabla u_l \approx 0$ leads to

$$\begin{aligned} \kappa_{P_1}^2(u_l) + \kappa_{P_2}^2(u_l) &= (\kappa_{P_1}(u_l) + \kappa_{P_2}(u_l))^2 - 2\kappa_{P_1}(u_l)\kappa_{P_2}(u_l) \\ &= (\kappa_M(u_l))^2 - 2K_G(u_l) = (\nabla\cdot(\frac{\nabla u_l}{\sqrt{1+|\nabla u_l|^2}}))^2 - 2(\frac{u_{l_{x_1x_1}}u_{l_{x_2x_2}} - u_{l_{x_1x_2}}^2}{(1+|\nabla u_l|^2)^2}) \\ &\approx (\widehat{\kappa}_M(u_l))^2 - 2\widehat{K}_G(u_l) = (\Delta u_l)^2 - 2(u_{l_{x_1x_1}}u_{l_{x_2x_2}} - u_{l_{x_1x_2}}^2), \end{aligned} \tag{3.36}$$

where $\widehat{K}_G(u_l) = (u_{l_{x_1x_1}} u_{l_{x_2x_2}} - u^2_{l_{x_1x_2}})$ is the approximation of $K_G(u_l)$, the Gaussian curvature of the surface of u_l.

3.4.5 Total variation regularisation

Total variation regularisation introduced by [51, 53, 115] is based on the so-called TV semi-norm of ∇u_l and given by

$$\mathcal{R}^{\beta\mathrm{TV}}(\boldsymbol{u}) = \sum_{l=1}^{2} \int_\Omega |\nabla u_l|_\beta \, d\mathbf{x} = \sum_{l=1}^{2} \int_\Omega \sqrt{u^2_{l_{x_1}} + u^2_{l_{x_2}} + \beta} d\mathbf{x}. \tag{3.37}$$

One can show that its Gâteaux derivative is given by

$$\delta\mathcal{R}^{\beta\mathrm{TV}}(\boldsymbol{u};\boldsymbol{v}) = \left\langle \nabla_{\boldsymbol{u}} \mathcal{R}^{\beta\mathrm{TV}}, \boldsymbol{\eta} \right\rangle_{\mathcal{H}}, \tag{3.38}$$

with

$$\nabla_{\boldsymbol{u}} \mathcal{R}^{\beta\mathrm{TV}} = \begin{pmatrix} -\nabla \cdot \left(\frac{\nabla u_1}{|\nabla u_1|_\beta}\right) \\ -\nabla \cdot \left(\frac{\nabla u_2}{|\nabla u_2|_\beta}\right) \end{pmatrix} \tag{3.39}$$

subject to the Neumann boundary condition $\langle \nabla u_l, \boldsymbol{n} \rangle_{\mathbb{R}^2} = 0$ on $\partial\Omega$. Here $\beta > 0$ is a small real parameter for avoiding non-differentiable at zero; see more details in [51, 53, 115, 118].

As is known, $\mathcal{R}^{\mathrm{elas}}$, $\mathcal{R}^{\mathrm{diff}}$, $\mathcal{R}^{\mathrm{FMcurv}}$, and $\mathcal{R}^{\mathrm{HWcurv}}$ produce globally smooth deformation fields; see [33, 46, 47, 48, 49, 79, 78, 73, 75, 74, 89, 91, 104, 131]. While they are useful for several applications, they become poor if discontinuities or steep gradients in the deformation fields are expected (e.g. resulting from multiple moving objects or partially occluded objects). In order to preserve discontinuities of the deformation field, $\mathcal{R}^{\beta\mathrm{TV}}$ helps to preserve piecewise constant smoothness, which is much weaker than those global smoothness of $\mathcal{R}^{\mathrm{elas}}$, $\mathcal{R}^{\mathrm{diff}}$, $\mathcal{R}^{\mathrm{FMcurv}}$, and $\mathcal{R}^{\mathrm{HWcurv}}$. However, $\mathcal{R}^{\beta\mathrm{TV}}$ may not be suitable for some particular image registration problems, which require deformation fields having very strong smoothing properties; see [51, 53, 142]. It is still a challenge to design a regularisation technique or deformation model $\mathcal{R}$ suitable for both smooth and non-smooth registration problems. This task will be one of our main contributions in this thesis; see Chapter 6 – 7 later.

3.5 General solution schemes

Classified by the order of its major ingredients, there are two main types of numerical schemes to compute a numerical solution of the minimisation problem (3.2) for a given regularisation parameter α. The first is the so-called *optimise-discretise* approach and the second is the so-called *discretise-optimise* approach. The main idea of the first approach is to compute the Euler-Lagrange equations in the continuous domain as discussed in the previous sections and then solve its discretised version on the corresponding discrete domain by a method of our choices, e.g. a so-called parabolic and elliptic approach. On the other hand the latter approach aims to discretise the joint functional $\mathcal{J}_\alpha$ and then solve the discrete minimisation problem by standard optimisation techniques, e.g. steepest descent or Newton-type methods.

3.5.1 The optimise-discretise approach

For the optimise-discretise approach, the main aim is to solve the Euler-Lagrange equation, which generally turns out to be a nonlinear system of PDEs:

$$\boldsymbol{f}(\boldsymbol{u}) + \alpha\mathcal{A}(\boldsymbol{u}) = 0 \tag{3.40}$$

subject to the appropriate boundary conditions. In other words, the approach aims to satisfy the necessary condition for a minimiser of the joint functional (3.2).

Note that on one hand the first term $\boldsymbol{f}$ (usually nonlinear) is related to the Gâteaux derivative of the particular similarity measure $\mathcal{D}$, which can be viewed as the *external forces* in leading similar regions of the images into correspondences. On the other hand, the second term $\mathcal{A}$, which is the partial differential operator (linear or nonlinear) and viewed as the *internal forces* (or constraints) resulting from the Gâteaux derivative of $\mathcal{R}$, is used to regularise the deformation field $\boldsymbol{u}$ and resist the external forces until the equilibrium state governed by the Euler-Lagrange equation (3.40) is archived.

There are various numerical techniques for solving (3.40). These techniques can be broadly divided into two main categories: the *parabolic* and *elliptic* approaches. A parabolic approach (also known as gradient descent or time marching approach) performs by introducing the artificial time variable t and then determining the steady state solution of the system of time-dependent PDEs, e.g. if $\boldsymbol{f}$ is nonlinear and $\mathcal{A}$ is linear, the semi-implicit scheme can be defined by

$$\frac{\boldsymbol{u}(t^{(k+1)}) - \boldsymbol{u}(t^{(k)})}{\tau} = \boldsymbol{f}(\boldsymbol{u}(t^{(k)})) + \alpha\mathcal{A}(\boldsymbol{u}(t^{(k+1)})) \tag{3.41}$$

where $k \in \mathbb{N}_0$, $\boldsymbol{u}(t) = \boldsymbol{u}(\mathbf{x}; t)$, and $\tau > 0$ denotes the time length used to discretise $\partial_t \boldsymbol{u}(t)$; see [46, 47, 48, 78, 73, 90, 91, 104, 131]. For an elliptic approach it performs by directly solving the nonlinear system of PDEs with a method of our choice, e.g. if both $\boldsymbol{f}$ and $\mathcal{A}$ are nonlinear, the fixed-point (FP) iteration of (3.40) can be defined by

$$\boldsymbol{f}(\boldsymbol{u}^{[\nu]}) + \alpha\mathcal{A}[\boldsymbol{u}^{[\nu]}](\boldsymbol{u}^{[\nu+1]}) = 0 \tag{3.42}$$

where both $\boldsymbol{f}$ and $\mathcal{A}$ are linearised at the current approximation $\boldsymbol{u}^{[\nu]}$ and $\nu \in \mathbb{N}_0$ denotes the FP step; see [53, 54, 76, 94, 145].

3.5.2 The discretise-optimise approach

In this section, we shall briefly give the main idea of the discretise-optimise approach based in the Newton-type schemes. To this end, let us first consider the discrete minimisation problem corresponding to (3.2):

$$\min_{\boldsymbol{u}} \{\widetilde{\mathcal{J}}_\alpha(\boldsymbol{u}) = \widetilde{\mathcal{D}}(\boldsymbol{u}) + \alpha\widetilde{\mathcal{R}}(\boldsymbol{u})\}. \tag{3.43}$$

The next step is to linearise $\widetilde{\mathcal{J}}_\alpha$ around the current approximation $\boldsymbol{u}^{(k)}$ ($k \in \mathbb{N}_0$) by the Taylor expansion given by

$$\widetilde{\mathcal{J}}_\alpha(\boldsymbol{u}^{(k)} + \delta\boldsymbol{u}^{(k)}) = \widetilde{\mathcal{J}}_\alpha(\boldsymbol{u}^{(k)}) + \mathbf{J}_{\widetilde{\mathcal{J}}_\alpha}(\boldsymbol{u}^{(k)})\delta\boldsymbol{u}^{(k)} + \frac{1}{2}(\delta\boldsymbol{u}^{(k)})^\top \mathbf{H}_{\widetilde{\mathcal{J}}_\alpha}(\boldsymbol{u}^{(k)})\delta\boldsymbol{u}^{(k)} \tag{3.44}$$

and define an (outer) iteration by

$$\boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k)} + \zeta^{(k)} \delta \boldsymbol{u}^{(k)}. \tag{3.45}$$

Here $\mathbf{J}_{\widetilde{\mathcal{J}}_\alpha}(\boldsymbol{u}^{(k)})$, $\mathbf{H}_{\widetilde{\mathcal{J}}_\alpha}(\boldsymbol{u}^{(k)})$ are the Jacobian and the Hessian of $\widetilde{\mathcal{J}}_\alpha$ at $\boldsymbol{u}^{(k)}$ and $\zeta^{(k)} > 0$ is the line-search parameter used to guarantee the reduction of $\widetilde{\mathcal{J}}_\alpha$ in each outer step k. For Newton-type methods, the perturbation $\delta \boldsymbol{u}^{(k)}$ is determined by solving the normal equation

$$\frac{1}{2}\widetilde{\mathbf{H}}_{\widetilde{\mathcal{J}}_\alpha}(\boldsymbol{u}^{(k)})\delta \boldsymbol{u}^{(k)} = -\mathbf{J}_{\widetilde{\mathcal{J}}_\alpha}(\boldsymbol{u}^{(k)}) \tag{3.46}$$

by a method of our choices (e.g. linear MG methods), considered as the inner step. Here $\widetilde{\mathbf{H}}_{\widetilde{\mathcal{J}}_\alpha}(\boldsymbol{u}^{(k)})$ is an approximate Hessian; see [60, 59, 65, 72, 76, 77, 83, 89].

Practically, no matter which method is used, both approaches are integrated with a so-called multi-resolution technique in order to provide reliable initial guesses, avoid getting in the trap of unwanted minimisers and save computational times [32, 86, 94, 99, 120, 130, 132].

3.6 A brief survey of existing multigrid methods

One of the main aims in this thesis is to propose efficient numerical methods for solving Euler-Lagrange equations as given by (3.40) consisting coupled and nonlinear PDEs and resulting from the variational formulation (3.2). Among fast iterative methods, multigrid approaches have been successfully used as fast registration algorithms for high-resolution digital images by offering the potential of optimal efficiency. They may classify into 2 categories:

1) **Linear multigrid framework**. Haber and Modersitzki [65] used the discretise-optimise framework by combining an inexact Gauss-Newton (GN) method with a linear multigrid method as a coupled outer-inner iteration method for solving the elastic image registration problem. Henn [73] considered the curvature image registration in the optimise-discretise framework and introduced a coupled outer-inner iteration method like (3.41) with the inner solver provided by a linear multigrid method for solving the system of the fourth-order linear PDEs. Hömke [83] concentrated on the elastic image registration and used the discretise-optimise approach based on a regularised GN method with a trust region approach in which one normal equation corresponding to a linear subproblem is solved iteratively with a linear multigrid method. Köstler et al. [89] introduced a combined diffusion- and curvature-based regulariser for optical flow and deformable image registration problems and solved the resulting minimisation problem represented in terms of (3.2) with the discretise-optimise framework by combining an inexact GN method with a linear multigrid approach. Stürmer et al. [131] considered the diffusion image registration in the optimise-discretise framework and solved the system of nonlinear PDEs using a coupled outer-inner iteration method like (3.41), where the inner iteration is solved by a linear multigrid method commonly used for heat equations.

2) **Nonlinear multigrid framework**. The use of the nonlinear multigrid (NMG) methods can be found in works of [53, 54, 76, 145]. In particular, Frohn-Schauf et al. [53] considered

the direct minimisation of the data term $\mathcal{D}^{\mathrm{SSD}}$ (3.7) in the Newton framework and applied the total variation (TV) regularisation at a Newton perturbation step; further they solved the resulting nonlinear system by the full approximation scheme nonlinear multigrid (FAS-NMG) method due to Brandt [11] with an augmentation method and a line relaxation smoother. Gao et al. [54] used the optimise-discretise framework to solve the diffusion image registration problem, where the full multigrid (FMG) method with the Newton-Gauss-Seidel smoother (i.e. global linearisation by Newton's iteration and Gauss-Seidel (GS) iteration for the resulting linear systems [94, 134]) is used to solve the system of nonlinear PDEs. Henn and Witsch [76] solved the elastic image registration problem in the optimise-discretise framework using a FAS-NMG method with the Jacobi smoother plus a line-search procedure to avoid effects on the regularisation parameter α. Finally, Zikic et al. [145] used the optimise-discretise framework with the FMG method to solve the diffusion image registration problem, where the system of nonlinear PDEs is solved by a fixed-point (FP) type of smoothers within the semi-implicit time marching approach.

We also remark that the 2D optical flow formulation (that does not use $\mathcal{D}^{\mathrm{SSD}}$) suitable for registering closely related images (e.g. video sequences) can be solved by multigrid techniques; see [19, 18].

Chapter 4

A Robust Affine Image Registration Method

As already pointed out in §1.1, rigid image registration alone cannot always provide a satisfactory registration result, particularly in many medical applications (e.g. one cannot ensure the patient sits in the identical position with respect to the equipment each time), while non-rigid or deformable image registration may not be quick enough for ready use.

This chapter is mainly concerned with affine image registration because it is applicable to a large class of deformable image registration methods by providing the good initial positions for the image to be registered. Moreover, as is well-known, an affine method is always many orders of magnitude faster than a variational image registration approach using non-parametric transformations due to much less unknowns involved; see [85, 86, 96, 104, 125, 132, 141, 143] and the references therein.

4.1 Introduction

For affine registration the SSD functional, which is viewed as the nonlinear least-squared (NLS) model, is usually applied when the image intensities of the given images are comparable. Although there are only 6 parameters for affine transformations, iterative methods to solve the underling nonlinear minimisation can suffer from convergence problems if good initial guesses are not possible (i.e. even after we attempt to devise good initial guesses). A theoretical reason may be that image registration problem is ill-posed in the sense of Hadamard. The information provided by the given images and the least-squared model are not sufficient to ensure the existence, uniqueness, and stability of a solution [76]. This motivates us to introduce regularisation into affine registration for constraining affine parameters, as one would do for other ill-posed problems [27, 104, 112, 137]. The result is a refined affine registration model that can be solved by converging methods for a large class of imaging problems.

The rest of the chapter is organized as follows. We introduce the affine and the diffusion image registration respectively in §4.2 and §4.3, and then present four methods to improve affine registration in §4.4. A regularised affine registration (RAR) model is presented in §4.5,

followed by a regularisation parameter selection algorithm in §4.6. Some numerical experiments on the performance of the proposed method are presented in §4.7 before conclusions in §4.8.

4.2 The preliminaries, affine image registration and solution methods

Assuming that in continuous variables the given images can be represented by the continuous mappings $R, T : \Omega \subset \mathbb{R}^2 \rightarrow V \subset \mathbb{R}_0^+$. It is customary to consider $\Omega = [0,1]^2$ and $V = [0,1]$ for gray-scale images. In practice, two discrete images of the same size $n_1 \times n_2$ are given: the reference R and the template T.

For each pixel $\mathbf{x} = (x_1, x_2)^\top$, denote by $\varphi = \varphi(\mathbf{x}) : \Omega \rightarrow \Omega$ the *unknown* coordinate transformation that produces the alignment between the reference R and the transformed version of the template

$$T = T \circ \varphi = T_\varphi(\mathbf{x}) = T(\varphi(\mathbf{x})). \tag{4.1}$$

We hope to achieve that $T \approx R$ or $T - R \approx 0$. Here the transformation φ has 2 components

$$\varphi(\mathbf{x}) = (\varphi_1(\mathbf{x}), \varphi_2(\mathbf{x}))^\top. \tag{4.2}$$

As already mentioned in §3.2, all registration strategies require a suitable similarity measure $\mathcal{D}$ in order to measure how similar these two images are under the transformation φ. This means that the general registration problem aim is to minimise this measure in order to determine φ:

$$\text{Find } \varphi = (\varphi_1, \varphi_2)^\top \text{ such that } \mathcal{D}[T_\varphi, R, \varphi] = \mathcal{D}[\varphi] \text{ is minimal.} \tag{4.3}$$

We adopt the SSD or the least-squared function $\mathcal{D}$ defined by

$$\mathcal{D}[T_\varphi, R, \varphi] = \frac{1}{2}\int_\Omega (T(\varphi(\mathbf{x})) - R(\mathbf{x}))^2 d\mathbf{x} = \frac{1}{2}\|T_\varphi - R\|_{L_2}^2 = \mathcal{D}[\varphi] \tag{4.4}$$

as the objective function, where $\|\cdot\|_{L_2}$ denotes the L_2-norm.

4.2.1 Affine transformation

Affine transformation is one of the most commonly used methods in registering two images; see [85, 86, 96, 104, 132, 141, 143] and references therein. Although only linear, it models a combination of effects stemming from four simple transformations: translating, rotating, scaling and shearing. An affine transformation corrects some global distortions in the images to be registered. In this section, we first introduce the model and then discuss two numerical methods for solving it.

An affine registration model assumes that the above transformation φ is linear i.e.

$$\varphi(\mathbf{x}) = \varphi_{\mathbf{a}}(\mathbf{x}) = \begin{bmatrix} \varphi_{\mathbf{a}_1}(\mathbf{x}) \\ \varphi_{\mathbf{a}_2}(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} a_1 & a_2 \\ a_4 & a_5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} a_3 \\ a_6 \end{bmatrix} = \mathbf{A}\mathbf{x} + \mathbf{b}, \tag{4.5}$$

where $\mathbf{A} = \begin{bmatrix} a_1 & a_2 \\ a_4 & a_5 \end{bmatrix}$ and $\mathbf{b} = \begin{bmatrix} a_3 \\ a_6 \end{bmatrix}$ are the affine transformation matrix and the translation vector respectively, for all $\mathbf{x} \in \Omega$. Here for optimisation purpose, the vector

$$\mathbf{a} = (a_1, a_2, a_3, a_4, a_5, a_6)^\top \in \mathbb{R}^6$$

will be used shortly. Clearly the inverse transform is simply $\mathbf{x} = \mathbf{A}^{-1}(\varphi_{\mathbf{a}} - \mathbf{b})$ if $\mathbf{A}$ is invertible. Note that $\mathbf{A}$ can be decomposed into a product of a rotation, a scaling, a shear in x_1- (and/or x_2-) direction or a combination of these simple transformations

$$\mathbf{A} = \begin{bmatrix} a_1 & a_2 \\ a_4 & a_5 \end{bmatrix} = \underbrace{\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}}_{\text{rotation}} \underbrace{\begin{bmatrix} s_{x_1} & 0 \\ 0 & s_{x_2} \end{bmatrix}}_{\text{scaling}} \underbrace{\begin{bmatrix} 1 & S_{x_1} \\ 0 & 1 \end{bmatrix}}_{\text{shear}} \tag{4.6}$$

where θ is the rotation angle, s_{x_1}, s_{x_2} are the scaling parameters, and S_{x_1} is the shear factor in x_1-direction. Clearly this kind of decomposition is not unique. It is clear that both a rigid-body transformation with $s_{x_1} = s_{x_2} = 1$ and $S_{x_1} = 0$ taking the form

$$\mathbf{A} = \begin{bmatrix} a_1 & a_2 \\ a_4 & a_5 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

and a similarity transformation with $0 < s_{x_1} = s_{x_2}$ and $S_{x_1} = 0$ taking the form

$$\mathbf{A} = \begin{bmatrix} a_1 & a_2 \\ a_4 & a_5 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} s_{x_1} & 0 \\ 0 & s_{x_2} \end{bmatrix}$$

are affine in special cases. From (4.4), the problem with such a $\varphi_{\mathbf{a}}$ is the affine image registration, formulated as follows:

$$\min_{\mathbf{a}\in\mathbb{R}^6} \mathcal{D}[\mathbf{a}] \tag{4.7}$$

where $\mathcal{D}[\mathbf{a}] = \mathcal{D}[\varphi_{\mathbf{a}}] = \frac{1}{2}\|T(\varphi_{\mathbf{a}}) - R\|_{L_2}^2 = \frac{1}{2}\|T(\mathbf{A}\mathbf{x} + \mathbf{b}) - R\|_{L_2}^2$.

Now consider how the registration problem (4.7) is solved by the so-called *discretise-optimise* approach as discussed in §3.5.2. Let $\mathbf{T}$ and $\mathbf{R}$ denote the discrete images of T and R in terms of $n_1 \times n_2$ arrays of image intensities. For ease of presentation, let $\mathbf{T}$ and $\mathbf{R}$, of dimension $N = n_1 n_2$, be pixel-wise ordered in a lexicographical order and denoted as follows:

$$\mathbf{T} = (t_{1,1}, t_{2,1}, ..., t_{i_1,i_2}, ..., t_{n_1,n_2})^\top \quad \text{and} \quad \mathbf{R} = (r_{1,1}, r_{2,1}, ..., r_{i_1,i_2}, ...r_{n_1,n_2})^\top, \tag{4.8}$$

where $1 \le i_1 \le n_1$ and $1 \le i_2 \le n_2$. Each element in the grid vectors $\mathbf{T}$ and $\mathbf{R}$ represents a pixel's gray intensity between black (0) and white (1). Given an affine transformation $\varphi_{\mathbf{a}} = (\varphi_{\mathbf{a}_1}, \varphi_{\mathbf{a}_2})^\top$, the discrete form of the transformed template image F can be expressed as:

$$\begin{aligned} F(\mathbf{a}) = (&t_{a_1+a_2+a_3,\ a_4+a_5+a_6}, t_{2a_1+a_2+a_3,\ 2a_4+a_5+a_6}, \cdots, \\ &t_{a_1 i_1+a_2 i_2+a_3,\ a_4 i_1+a_5 i_2+a_6}, \cdots, t_{a_1 n_1+a_2 n_2+a_3,\ a_4 n_1+a_5 n_2+a_6})^\top, \end{aligned} \tag{4.9}$$

where $F : \mathbb{R}^6 \to \mathbb{R}^N$. Then minimisation problem (4.7) is equivalent to the following

$$\min_{\mathbf{a}\in\mathbb{R}^6} \mathcal{D}[\mathbf{a}]\, N = \frac{1}{2}||F(\mathbf{a}) - \mathbf{R}||_2^2 = \frac{1}{2}||\mathbf{d}(\mathbf{a})||_2^2 = \frac{1}{2}\sum_{i=1}^{N} d_i^2(\mathbf{a}), \tag{4.10}$$

where the factor $N = n_1 n_2 = 1/(h_1 h_2)$ due to the discretisation procedure with h_1, h_2 (the spatial mesh lengths) can be ignored here but will be used later in §4.5, and $\mathbf{d}(\mathbf{a}) = F(\mathbf{a}) - \mathbf{R} \in \mathbb{R}^N$ is the so-called *residual vector*. The first order condition of (4.10) is

$$\mathbf{g}(\mathbf{a}) = \nabla_{\mathbf{a}}\mathcal{D}[\mathbf{a}] = \mathbf{J}^\top(\mathbf{a})(F(\mathbf{a}) - \mathbf{R}) = \mathbf{J}^\top \mathbf{d}(\mathbf{a}) = \mathbf{0}, \tag{4.11}$$

where $\mathbf{g}(\mathbf{a}) \in \mathbb{R}^6$ and the Jacobian matrix $\mathbf{J}$ given by

$$\mathbf{J}_{i,j} = \frac{\partial d_i}{\partial a_j} \quad \text{for } 1 \leq i \leq N, 1 \leq j \leq 6. \tag{4.12}$$

Solving (4.11) for $\mathbf{a}$ is a nonlinear problem and its solution requires an iterative approach. Let $\mathbf{a}^{(k)}$ be $\mathbf{a}$ at the kth iteration. Here, we must find a perturbation $\delta\mathbf{a}^{(k)}$ first and then update the solution vector by

$$\mathbf{a}^{(k+1)} = \mathbf{a}^{(k)} + \delta\mathbf{a}^{(k)}. \tag{4.13}$$

For a full Newton method, the perturbation $\delta\mathbf{a}^{(k)}$ is determined by solving

$$\mathbf{H}(\mathbf{a}^{(k)})\delta\mathbf{a}^{(k)} = -\mathbf{g}(\mathbf{a}^{(k)}), \tag{4.14}$$

where the Hessian of $\mathcal{D}$ is denoted by

$$\mathbf{H}(\mathbf{a}) = \mathbf{J}^\top(\mathbf{a})\mathbf{J}(\mathbf{a}) + \sum_{i=1}^{N} d_i(\mathbf{a})\nabla^2 d_i(\mathbf{a}). \tag{4.15}$$

As pointed by [104, p.79], this Newton method may be not suitable in registering two images for practical applications because computing higher order derivatives is time consuming and numerically unstable. In order to improve on the Newton method, we can take advantages of the particular structure of $\mathbf{H}$ to design a Newton variant to compute $\delta\mathbf{a}^{(k)}$.

4.2.2 The Gauss-Newton (GN) method

Note that the Hessian matrix is precisely $\mathbf{H}(\mathbf{a}) = \mathbf{J}^\top(\mathbf{a})\mathbf{J}(\mathbf{a})$ if $d_i = 0$ for all i (i.e. the residuals are zero at the solution $\mathbf{a}^*$) or if $\nabla^2 d_i(\mathbf{a}) = 0$ when d_i is a linear function of $\mathbf{a}$. This suggests that in other cases the Hessian matrix may also be approximated by this formula [129]. The resulting approximation leads to the Gauss-Newton (GN) method, defined by

$$\widetilde{\mathbf{H}}(\mathbf{a}^{(k)})\delta\mathbf{a}^{(k)} = -\mathbf{g}(\mathbf{a}^{(k)}), \tag{4.16}$$

where one uses the matrix $\widetilde{\mathbf{H}}(\mathbf{a}^{(k)}) = \mathbf{J}^\top(\mathbf{a}^{(k)})\mathbf{J}(\mathbf{a}^{(k)})$ to approximate $\mathbf{H}(\mathbf{a}^{(k)})$.

The above GN method requires damping to ensure convergence, because we may not be able to provide a good initial solution, close to a minimum of $\mathcal{D}$. The damped GN method can be generated by

$$\mathbf{a}^{(k+1)} = \mathbf{a}^{(k)} + \zeta^{(k)}\delta\mathbf{a}^{(k)} \tag{4.17}$$

where the positive scalar $\zeta^{(k)}$ is the so-called *line-search* parameter used to ensure that a GN step adequately reduces $\mathcal{D}$ and to rule out an unacceptable short step. More precisely, $\zeta^{(k)}$ is determined by

$$\zeta^{(k)} = \min_{\zeta} \mathcal{D}(\mathbf{a}^{(k)} + \zeta\delta\mathbf{a}^{(k)}).$$

Solving this line-search problem is by a *backtracking* algorithm which begins with $\zeta^{(k)} = 1$, and then, if $\mathbf{a}^{(k)} + \delta\mathbf{a}^{(k)}$ is not acceptable, reduces $\alpha^{(k)}$ until an acceptable $\mathbf{a}^{(k)} + \zeta^{(k)}\delta\mathbf{a}^{(k)}$ is found. The acceptability is decided by the so-called Wolfe or Armijo-Goldstein conditions safeguarding upper and lower bounds; see [39, 50, 87, 109].

4.2.3 The Levenberg-Marquardt (LM) method

The GN method (4.16) assumes that $\widetilde{\mathbf{H}}(\mathbf{a}^{(k)})$ is well-conditioned or at least non-singular. To remove this assumption, an improved formulation by adding a positive multiple of the identity matrix $\mathbf{I}$, $\widetilde{\mathbf{H}}(\mathbf{a}^{(k)})= \left[\mathbf{J}^\top(\mathbf{a}^{(k)})\mathbf{J}(\mathbf{a}^{(k)}) + \mu^{(k)}\mathbf{I}\right]$, is the Levenberg-Marquardt (LM) method:

$$[\mathbf{J}^\top(\mathbf{a}^{(k)})\mathbf{J}(\mathbf{a}^{(k)}) + \mu^{(k)}\mathbf{I}]\delta\mathbf{a}^{(k)} = -\mathbf{g}(\mathbf{a}^{(k)}). \tag{4.18}$$

At iteration k, the positive LM parameter $\mu^{(k)}$ is adjusted to guarantee that the search direction $\delta\mathbf{a}^{(k)}$ in (4.18) is a descent direction[1]. Then we obtain a steepest descent direction for large $\mu^{(k)}$, when the current iterate is far from the solution. On the other hand, this descent direction is approximately a GN search direction for small $\mu^{(k)}$, when the iterates get close enough to the solution. Using a frame work of trust region strategies, the LM parameter $\mu^{(k)}$ is determined in such a way that

$$||\delta\mathbf{a}^{(k)}||_2^2 = ||[\mathbf{J}^\top(\mathbf{a}^{(k)})\mathbf{J}(\mathbf{a}^{(k)}) + \mu^{(k)}\mathbf{I}]^{-1}[-\mathbf{g}(\mathbf{a}^{(k)})]||_2^2 \leq \eta^{(k)} \tag{4.19}$$

where $\eta^{(k)} \geq 0$ is a prescribed trust region radius. A new LM step is then generated by $\mathbf{a}^{(k+1)} = \mathbf{a}^{(k)} + \delta\mathbf{a}^{(k)}$. As remarked by [41, 42, 70, 72], this numerical scheme is related to Tikhonov regularisation (see §4.5 later) and is sometimes called the regularising Levenberg-Marquardt method as shown in (4.43).

4.2.4 Some registration results using the GN and LM methods

In this section, some registration results using the GN method (§4.2.2) and the LM method (§4.2.3) are presented, to illustrate the non-robustness of both GN and LM methods.

Two examples are provided with the first one to show that both methods are capable of correctly registering 2 images and the second one to show that both methods can fail to converge to an acceptable solution, i.e. fail to register 2 images (in particular our examples will differ in outliers). In both examples, the images are of size 128×128 and for both GN and LM, we use the termination criterion $\|\delta\mathbf{a}\|_2 \leq \epsilon = 10^{-6}$ within the maximum of iterations IMAX= 300. The bilinear interpolation technique was applied in all examples for computing the transformed template image $F(\mathbf{a}) = \mathbf{T}_{\varphi_\mathbf{a}}$. Here the relative residual is used as the error indicator: $error = \|\mathbf{g}(\mathbf{a}^{(k)})\|_2$.

Example 4.2.1 (A successful case) *We consider the registration problem for a pair of MR images of a human head*[2]*, with the reference image* $\mathbf{R}$ *and the template image* $\mathbf{T}$ *respectively in Figure* 4.1 *(a) – (b). Using the initial guess* $\mathbf{a}^{(0)} = (1,0,0,0,1,0)^\top$ *(i.e. we start with* $\varphi_0(\mathbf{x}) = \mathbf{x}$*), both the GN and LM methods can successfully register this example as shown respectively in Figure* 4.1 *(c) – (d).*

Example 4.2.2 (An unsuccessful case) *Here we consider another pair of MR images (similar to Example 4.2.1), as shown in Figure* 4.2 *(a)–(b), where* $\mathbf{T}$ *contains tumor like circles. As*

[1] $\delta\mathbf{a}$ is a descent direction if $\delta\mathbf{a}^\top\mathbf{g}(\mathbf{a}) < 0$.

[2] Source: http://www.cis.rit.edu/class/schp730/lect/lect-1.htm

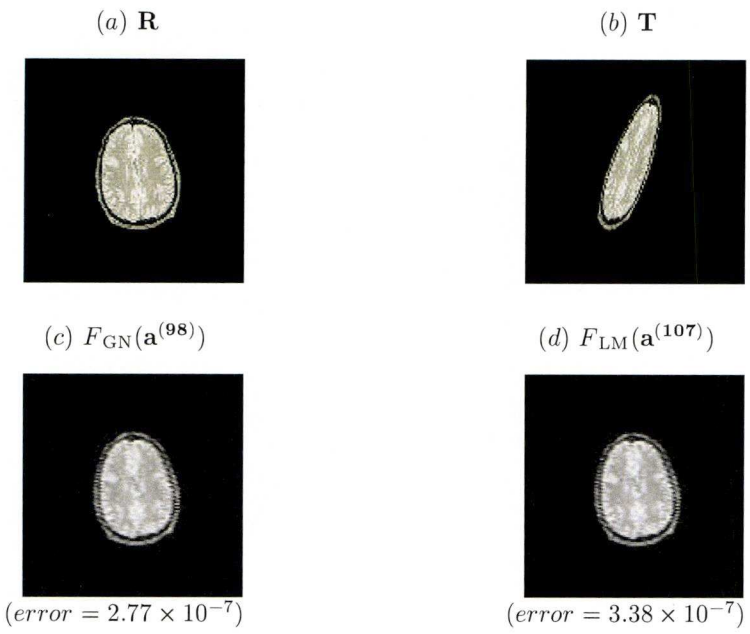


Figure 4.1: Example 4.2.1: Successful registration results of the MR images of a human head. The first row shows the reference image $\mathbf{R}$ (a), the template image $\mathbf{T}$ (b). The second row presents the registered images $F_{\mathrm{GN}}(\mathbf{a}^{(98)})$ (c) and $F_{\mathrm{LM}}(\mathbf{a}^{(107)})$ (d) obtained from using the GN and LM methods, respectively.

in Example 4.2.1, *the initial guess solution* $\mathbf{a}^{(0)} = (1, 0, 0, 0, 1, 0)^\top$ *is used. It turns out that both GN and LM methods get stuck (at a local minimum of* $\mathcal{D}$*) and fail to obtain correctly converged solutions, as shown in Figure* 4.2 $(a) - (b)$. *Here we are certain about reaching a local minimum because the residual error is small, and the registration failure because we can observe the large visual difference between* $F(\mathbf{a})$ *and* $\mathbf{R}$ *(i.e. the matching error is not the smallest possible).*

Based on Examples 4.2.1 and 4.2.2 and other tests, we confirm that (as is known) both methods are not robust enough as their convergence strongly depends on initial guess solutions. Various ways of finding good initial guess solutions will be discussed shortly in §4.4.

4.3 Deformable image registration

Having discussed a parametric registration model, we now give a brief review of a non-parametric model – the variational diffusion model for deformable registration [46, 104]. We shall show that, although the nonlinear multigrid method [11, 12, 134, 139, 140] is effective in solving the model, an affine pre-registration step can further speed up the solution. Hence it is of interest to look for reliable affine methods. We first review the general Tikhonov regularisation idea [27, 104, 112, 137].

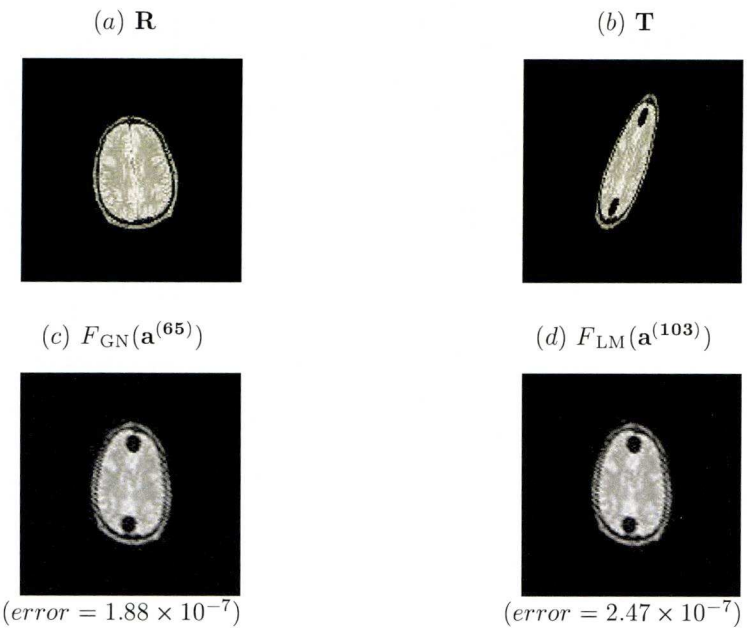


Figure 4.2: Example 4.2.2: Unsuccessful registration results of the MR images of a human head. The first row shows the reference image $\mathbf{R}$ (a), the template image $\mathbf{T}$ (b). The second row presents the registered images $F_{\mathrm{GN}}(\mathbf{a}^{(65)})$ (c) and $F_{\mathrm{LM}}(\mathbf{a}^{(103)})$ (d) obtained from using the GN and LM methods, respectively.

4.3.1 Variational approach

As an inverse problem, the general registration problem (4.3) denoted by $\min_{\varphi} \mathcal{D}[\varphi]$ is ill-posed and can be converted to a well-posed problem by Tikhonov regularisation leading to

$$\min_{\varphi}\{\mathcal{J}_\alpha(\varphi) = \mathcal{D}(\varphi) + \alpha\mathcal{R}(\mathbf{x} - \varphi)\} \tag{4.20}$$

where the positive regulariser $\mathcal{R}$ may be chosen differently [104], and $\alpha > 0$ is the regularisation parameter, which controls the fitting of the registered image, as measured by the first term $\mathcal{D}(\varphi)$, and the regularity of the solution, as measured by the second term $\mathcal{R}(\mathbf{x} - \varphi)$.

To have a consistent notation with the original idea for the diffusion image registration, we define the new deformation variable $\boldsymbol{u}(\mathbf{x}) = \mathbf{x} - \varphi(\mathbf{x})$, and then the geometric transformation $\varphi = \varphi(\mathbf{x})$ depends on the deformation field $\boldsymbol{u} = \boldsymbol{u}(\mathbf{x})$. As mentioned in [104, p.77], the problem of finding the transformation $\varphi = \varphi(\mathbf{x}) = \mathbf{x} - \boldsymbol{u}(\mathbf{x})$ and the deformation field $\boldsymbol{u} = \boldsymbol{u}(\mathbf{x}) = \mathbf{x} - \varphi(\mathbf{x})$ represented by (4.20) is equivalent. Then the variational problem (4.20) becomes

$$\min_{\boldsymbol{u}}\{\mathcal{J}_\alpha(\boldsymbol{u}) = \mathcal{D}(\boldsymbol{u}) + \alpha\mathcal{R}(\boldsymbol{u})\}, \tag{4.21}$$

where $\mathcal{D}(\boldsymbol{u}) = \frac{1}{2}\int_\Omega \left(T\left(\mathbf{x} - \boldsymbol{u}(\mathbf{x})\right) - R(\mathbf{x})\right)^2 d\mathbf{x}$.

4.3.2 Diffusion image registration

The diffusion image registration introduced by Fischer and Modersitzki [46] chose the following *diffusive regulariser*

$$\mathcal{R}(\boldsymbol{u}) = \frac{1}{2}\int_{\Omega}(|\nabla u_1|^2 + |\nabla u_2|^2)d\mathbf{x}, \tag{4.22}$$

subject to Neumann boundary conditions, i.e.,

$$\frac{\partial u_\ell}{\partial \boldsymbol{n}} = 0 \quad \text{for } \mathbf{x} \in \partial\Omega \text{ and } \ell = 1, 2. \tag{4.23}$$

Here, $\boldsymbol{n}$ denotes the unit outer normal vector on $\partial\Omega$. The Euler-Lagrange equation for the variational problem (4.21) is the following

$$-\alpha\Delta\boldsymbol{u}(\mathbf{x}) - (T(\mathbf{x} - \boldsymbol{u}(\mathbf{x})) - R(\mathbf{x})) \cdot \nabla_{\boldsymbol{u}} T(\mathbf{x} - \boldsymbol{u}(\mathbf{x})) = 0, \quad \mathbf{x} \in \Omega, \tag{4.24}$$

where the Gâteaux-derivatives of $\mathcal{D}$ and $\mathcal{R}$ are used and Δ denotes the Laplace operator with $\Delta\boldsymbol{u}(\mathbf{x}) = (\Delta u_1(\mathbf{x}), \Delta u_2(\mathbf{x}))^\top$. Note that (4.24) denotes a system of two nonlinear PDEs.

4.3.3 Numerical treatment and results

In [46], the cell-centered finite difference scheme is recommended to discretise the parabolic version of (4.24) i.e.

$$\frac{\partial \boldsymbol{u}}{\partial t} = \alpha\Delta\boldsymbol{u}(\mathbf{x}) + (T(\mathbf{x} - \boldsymbol{u}(\mathbf{x})) - R(\mathbf{x})) \cdot \nabla_{\boldsymbol{u}} T(\mathbf{x} - \boldsymbol{u}(\mathbf{x}))$$

and solve the discrete system by the so-called additive operator splitting (AOS) method which is a semi-implicit time marching method.

Here the finite difference method is first applied with (4.24), followed by some results from a full approximation scheme nonlinear multigrid (with full multigrid initialisation) method, denoted by FAS-FMG, as in [11, 12, 134, 139, 140]. The basic steps are briefly summarised as follows: (i) Convert the original fine grid problem to a hierarchy of coarser levels with standard coarsening. The linearised Guass-Seidel smoother consisting of outer and inner iterations is employed for (4.24), while on the coarsest level the AOS-scheme of [46] is used. We take the number of pre- and post-smoothing (outer) steps to be 3, and the number of inner iterations to be 2. (ii) Use the standard bi-linear interpolation and restriction operators; see the coming chapter for more details.

Example 4.3.1 *We consider the deformable registration problem of the X-Ray images of a human hand*[3]*. Figures* 4.3 $(a) - (b)$ *show the reference* $\mathbf{R}$ *and template* $\mathbf{T}$ *images. Clearly one can tell that the two images are not related by affine transforms. However we use an affine transform to provide a good initial guess which we denote by* $\mathbf{T}^{\mathbf{lin}}_{GN}$ *in Figure* 4.3 (c)*, obtained from the affine method as in* §4.2.2*.*

Then the registered images $F(\boldsymbol{u})$ *obtained from* (4.24) *with the FAS-FMG method with and without the affine pre-registration step are shown, respectively, in Figures* 4.3 $(e) - (f)$*. The*

[3]Source: http://www.math.mu-luebeck.de/safir/

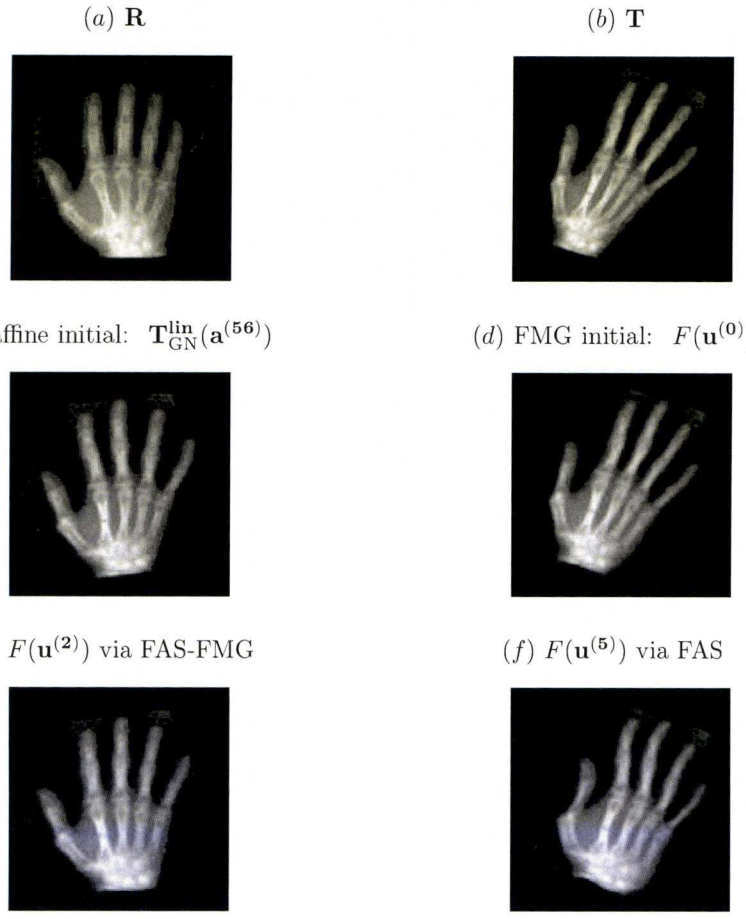


Figure 4.3: Example 4.3.1: Deformable registration results of the X-Ray images of a human hand, showing the importance of a pre-registration step. Left: (*a*) Reference **R**, (*c*) the linearly registered template (initial template) using the GN method $\mathbf{T}_{\mathrm{GN}}^{\mathrm{lin}}(\mathbf{a}^{(56)})$, and (*e*) the registered image $F(\mathbf{u}^{(2)})$ by FMG-FAS with (*c*). Right: (*b*) Template **T**, (*d*) the initial image $F(\mathbf{u}^{(0)})$ after FMG step, and (*f*) the (failed) registered image $F(\mathbf{u}^{(5)})$ with (*d*).

latter method (without using the affine pre-registration step) is not only much slower than the former with the affine step (only 2 *FAS cycles), but also it failed to register properly Figures* 4.3 (*f*). *Here we remark that without the affine pre-registration step, essentially, it is the FMG method that struggles on the coarsest grid.*

Through the above example, we see that a deformable registration approach can benefit from an affine pre-registration step whose convergence is of course of importance.

4.4 Techniques to improve affine registration methods

The convergence of both GN and LM methods depends on suitable initial guesses, as shown in §4.2.4. This section reviews first some existing methods that can provide a better initial guess

than the simple $\mathbf{a}^{(0)} = (1,0,0,0,1,0)^\top$ for the GN affine model (4.16), and then shows their numerical tests for both GN and LM methods.

4.4.1 Method 1 – Approximation based on image centers

Each of the two input images has a center location, defined by the pixel gray levels (or features dependent). If the two centers are quite different, re-positioning the center will give the affine registration problem a good initial guess for translation, i.e. the vector given by $\mathbf{a}^{(0)} = (1,0,a_3^{(0)},0,1,a_6^{(0)})^\top$, where $a_3^{(0)}$ and $a_6^{(0)}$ denote the re-positioning information.

This method can be summarised as follows:

(i) Estimate the centers c^T, c^R of the two input images T, R respectively:

$$\begin{cases} c^T = \begin{bmatrix} c_1^T \\ c_2^T \end{bmatrix} = \frac{\int_\Omega \mathbf{x}T(\mathbf{x})d\mathbf{x}}{\int_\Omega T(\mathbf{x})d\mathbf{x}} = \frac{(\sum_{i=1}^{n_1}\sum_{j=1}^{n_2} t_{i,j}i,\ \sum_{i=1}^{n_1}\sum_{j=1}^{n_2} t_{i,j}j)}{\sum_{i=1}^{n_1}\sum_{j=1}^{n_2} t_{i,j}}, \\ c^R = \begin{bmatrix} c_1^R \\ c_2^R \end{bmatrix} = \frac{\int_\Omega \mathbf{x}R(\mathbf{x})d\mathbf{x}}{\int_\Omega R(\mathbf{x})d\mathbf{x}} = \frac{(\sum_{i=1}^{n_1}\sum_{j=1}^{n_2} r_{i,j}i,\ \sum_{i=1}^{n_1}\sum_{j=1}^{n_2} r_{i,j}j)}{\sum_{i=1}^{n_1}\sum_{j=1}^{n_2} r_{i,j}} \equiv \mathbb{E}_R[\mathbf{x}]. \end{cases} \tag{4.25}$$

(ii) From the center differences, set $a_3^{(0)} = c_1^R - c_1^T$, $a_6^{(0)} = c_2^R - c_2^T$.

4.4.2 Method 2 – Approximation based on the rigid-body model

The next idea of providing a good initial guess for (4.7) is to reduce the number of parameters: assume there exists a rigid transformation between T and R. Then we have a parametric model $\varphi(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{b}$ with only 3 parameters (see (4.6)):

$$\mathbf{A} = \begin{bmatrix} a_1 & a_2 \\ a_4 & a_5 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} a_3 \\ a_6 \end{bmatrix}. \tag{4.26}$$

Here the above Method 1 could be used to initialise $\mathbf{b}$ while setting $\theta^{(0)} = 0$. Once this model is solved, the coefficients of $\mathbf{a}^{(0)}$ will be updated for the affine model.

4.4.3 Method 3 – Approximation based on principal axes transformation

The *principal axes transformation* (PAT) method was introduced to image processing by Hu since 1962 (see [104, 125] and references therein). It is an approximate registration approach using statistical features, the image center and an eigen decomposition of the covariance matrix, derived from the input images. Define the 2×2 covariance matrix of an image I by

$$\mathrm{Cov}_I = \mathbb{E}_I[(\mathbf{x}-c^I)(\mathbf{x}-c^I)^\top], \tag{4.27}$$

where c^I is the image center defined by (4.25). Since matrix Cov_I is real, symmetric, and positive semi-definite, it permits an eigenvalue decomposition [104]

$$\mathrm{Cov}_I = D(\rho_I)\Sigma_I^2 D(-\rho_I), \quad D(\rho_I) = \begin{bmatrix} \cos\rho_I & -\sin\rho_I \\ \sin\rho_I & \cos\rho_I \end{bmatrix}, \quad \Sigma_I = \begin{bmatrix} \sigma_{I,1} & 0 \\ 0 & \sigma_{I,2} \end{bmatrix} \tag{4.28}$$

where $D(\rho_I)$ denotes a rotation matrix, Σ_I is a scaling matrix, and $\sigma_{I,1}$ and $\sigma_{I,2}$ are standard deviations.

Now for images R, T, let c^R and c^T be the centers from (4.25). Then the following will be the approximate coefficients for an affine transform

$$\mathbf{A} = D(\rho_T)\,\Sigma_T \Sigma_R^{-1} D(\rho_R)^\top, \quad \mathbf{b} = c^T - \mathbf{A}c^R. \tag{4.29}$$

Finally the coefficients from (4.29) will be used to initialise $\mathbf{a}^{(0)}$ for the affine model.

4.4.4 Method 4 – Multi-resolution approach

Multi-resolution strategy is commonly used to provide reliable initial guesses for registration algorithms [86, 94, 99, 120, 130, 132]. The idea is to register the coarse resolution (low) images first and then interpolate the coarse solutions level by level to the finest resolution (high). The basic idea is essentially the same as a full multigrid method as in [11, 12, 134, 139, 140] and done in §4.3.3.

Suppose that we operate with L levels in total (using standard coarsening [134]), with $\ell = 1$ the coarsest level and $\ell = L$ the finest level. Here the size of the coarsest level 1 is chosen as 32×32 or 64×64, and the bi-linear interpolation is used. Although the full weighting operator may be used for restriction, the usual practice is to use a Gaussian-like kernel typically consisted of a 5×5 template of weights as follows. Take the reference image $\mathbf{R} = \mathbf{R}_L$ as example. Define a coarsening operation from $\mathbf{R}_\ell$ to $\mathbf{R}_{\ell-1}$, i.e. $\mathbf{R}_{\ell-1} = coarsen(\mathbf{R}_\ell)$, by

$$R_{\ell-1}(i,j) = \sum_{k_1=-2}^{2} \sum_{k_2=-2}^{2} w(k_1) w(k_2) R_\ell(2i + k_1, 2j + k_2),$$

where $w(0) = 2/5$, $w(\pm 1) = 1/4$, and $w(\pm 2) = 1/4 - w(0)/2$. On level ℓ a standard NLS method (either GN or LM) is used to compute the affine transformation up to some tolerance (e.g. $tol = 10^{-2}$), which is denoted by $\mathbf{a}_\ell \leftarrow Solver\_Step(\mathbf{T}_\ell, \mathbf{R}_\ell, \mathbf{a}_\ell)$. Then the whole procedure of Method 4 may be denoted by $\mathbf{a}_L \leftarrow multiresolution(\mathbf{T}_L, \mathbf{R}_L, \mathbf{a}_L, L)$ with a recursion step summarised below:

Algorithm 4.4.1 (Multi-resolution approach)
Implement $\mathbf{a}_\ell \leftarrow multiresolution(\mathbf{T}_\ell, \mathbf{R}_\ell, \mathbf{a}_\ell, \ell)$ *as follows:*

- If $\ell = 1$
  - Set $\mathbf{a}_\ell = (1, 0, 0, 0, 1, 0)^\top$ or use Methods 1-3 to work out an initial $\mathbf{a}_\ell$,
  - $\mathbf{a}_\ell \leftarrow Solver\_Step(\mathbf{T}_\ell, \mathbf{R}_\ell, \mathbf{a}_\ell)$.
- Else
  - $\mathbf{T}_{\ell-1} = coarsen(\mathbf{T}_\ell)$, $\mathbf{R}_{\ell-1} = coarsen(\mathbf{R}_\ell)$.
  - $\mathbf{a}_{\ell-1} \leftarrow multiresolution(\mathbf{T}_{\ell-1}, \mathbf{R}_{\ell-1}, \mathbf{a}_{\ell-1}, \ell - 1)$
  - $\mathbf{a}_\ell \leftarrow interpolate(\mathbf{a}_{\ell-1})$ as follows:
    $a_{i_\ell} = a_{i_{\ell-1}}$ for $i = 1, 2, 4, 5$ (the elements of the affine transformation matrix) and
    $a_{i_\ell} = 2a_{i_{\ell-1}}$ for $i = 3, 6$ (the elements of the translation matrix).
  - $\mathbf{a}_\ell \leftarrow Solver\_Step(\mathbf{T}_\ell, \mathbf{R}_\ell, \mathbf{a}_\ell)$.

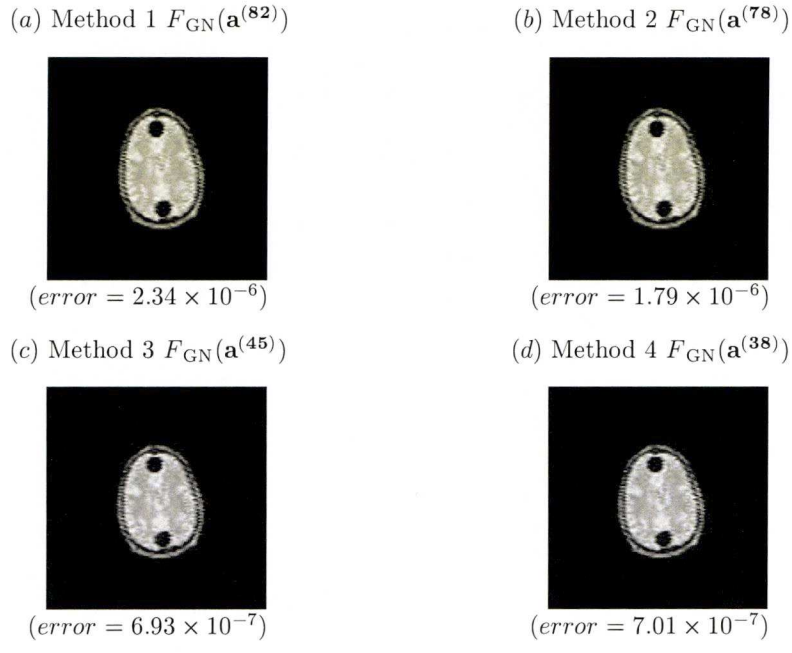


Figure 4.4: Example 4.2.2 re-solved: Correct registration results using the GN method with Methods $1-4$ providing initial guess solutions respectively for (a), (b), (c) and (d).

4.4.5 Applications of Methods $1-4$ to GN and LM methods

To illustrate the performance of Methods $1-4$, we give two successful examples: firstly re-solve Example 4.2.2 and secondly consider a new Example 4.4.1.

Recall from Figure 4.2 that both GN and LM methods failed to converge to the desirable solution for Example 4.2.2 with a simple initial guess. Now with Methods $1-4$ to provide initial guesses, both GN and LM methods work successfully – we show the registered results from GN in Figure 4.4 (while the LM results are virtually identical).

Example 4.4.1 *Here we consider the deformable registration problem for a pair of MR images of a human head, with Figure 4.5 (a)–(b) showing the reference image* $\mathbf{R}$ *and the template image* $\mathbf{T}$ *in size* 128×128.

As this is a deformable (not affine) problem, we can only use Methods $1-4$ *to provide an initial guess solution for the affine model, whose solution is then used for the diffusion registration method of* §4.3.3. *Figure* 4.5 $(c)-(d)$ *shows the results of affine GN and LM methods with Method* 4 *providing the initial guess, with the GN taking only* 5 *iterations and the LM taking* 11 *iterations on the finest resolution. Further, using Figure* 4.5 $(c)-(d)$ *as initial guesses, the diffusion registration method of* §4.3.3 *with* $\alpha = 0.058$ *gives the respectively registered images as depicted in Figure* 4.5 $(e)-(f)$.

Of the four methods, Method 4 is believed to be the best because Methods $1-2$ are not as general as $3-4$, and Method 3 is unable to resolve shear components [125]. However, even

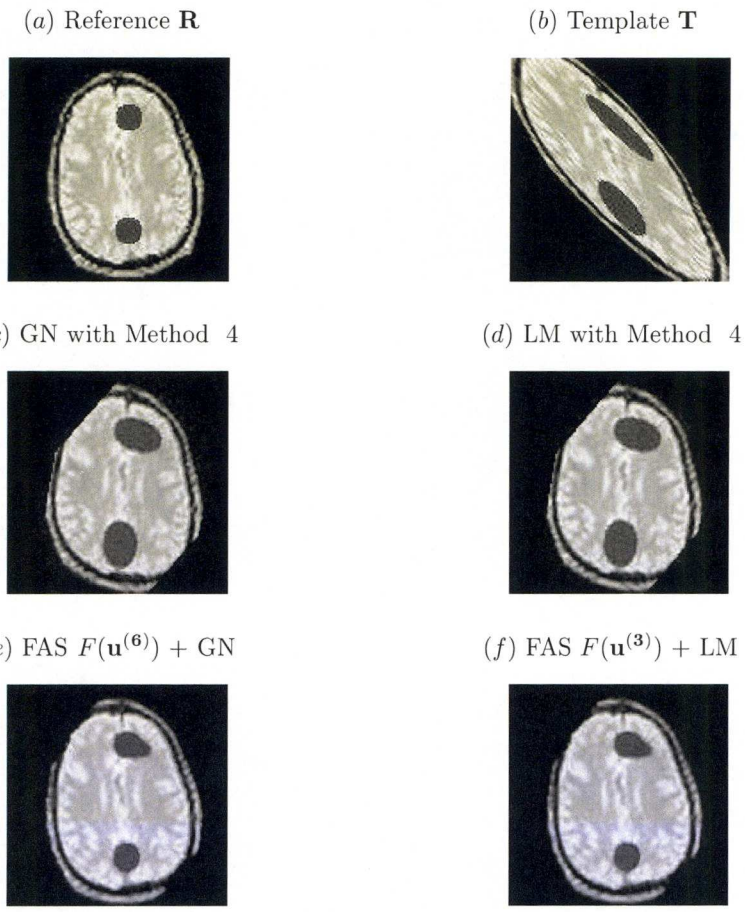


Figure 4.5: Example 4.4.1: Correct registration results of the MR images of a human head (deformable model of §4.3.2 with initial solutions provided by Method 4) as in row 1. The second row displays the helpful pre-registration images obtained from (c) the GN method $\mathbf{T}_{\text{GN}}$ and (d) the LM method $\mathbf{T}_{\text{LM}}$. The last row $(e)-(f)$ shows the deformable model (via FAS) registered images starting with $(c)-(d)$ respectively.

Method 4 cannot provide good initial guesses for some examples, as shown later in Example 4.7.2. Although one can think about designing better ways than Methods $1-4$ to provide more reliable and robust initial guesses for the affine model, our idea below is to propose a modified affine model that is less demanding than the standard model for initial guesses.

4.5 A regularised affine registration model

We propose to regularise the minimising functional. Although the regularisation idea is widely known for non-parametric transformation models (as in §4.3.1), it is not usually applied to parametric registration problems because the number of unknowns is relatively small compared with those of non-parametric transformations.

Motivated by (4.20), we solve, instead of (4.3), the following minimisation problem:

$$\min_{\varphi_{\mathbf{a}}}\{\mathcal{J}_\alpha(\varphi_{\mathbf{a}}) = N\mathcal{D}(\varphi_{\mathbf{a}}) + \overline{\alpha}\mathcal{R}(\mathbf{x}-\varphi_{\mathbf{a}})\}, \tag{4.30}$$

where the regulariser $\mathcal{R}$ for affine image registration is proposed to take the form

$$\mathcal{R}(x-\varphi_{\mathbf{a}}) = \begin{cases} \mathcal{R}_1 = \frac{1}{2}\sum_{i=1}^{2}\|x_i - \varphi_{\mathbf{a}_i}(\mathbf{x})\|^2_{H^1_{semi}} = \frac{1}{2}\sum_{i=1}^{2}\int_\Omega(|\nabla_{x_i}(x_i-\varphi_{\mathbf{a}_i}(\mathbf{x}))|^2_{L_2})d\Omega, \\ \mathcal{R}_2 = \frac{1}{2}\sum_{i=1}^{2}\|x_i - \varphi_{\mathbf{a}_i}(\mathbf{x})\|^2_{L_2} = \frac{1}{2}\sum_{i=1}^{2}\int_\Omega((x_i-\varphi_{\mathbf{a}_i}(\mathbf{x}))^2)d\Omega, \\ \mathcal{R}_3 = \mathcal{R}_1[\mathbf{x}-\varphi_{\mathbf{a}}] + \mathcal{R}_2[\mathbf{x}-\varphi_{\mathbf{a}}], \\ \mathcal{R}_4 = \frac{1}{2}\|\mathbf{a}\|_2^2. \end{cases} \tag{4.31}$$

Here the regularisers $\mathcal{R}_1$, $\mathcal{R}_2$, and $\mathcal{R}_3$ are motivated by regularisation, differing only in norms for functions, which are respectively the Sobolev semi-norm H^1_{semi}, L_2-norm, and Sobolev norm H^1. $\mathcal{R}_4$ is a simple option, using the 2−norm for $\mathbf{a}$. Clearly the new regularised affine registration (RAR) model (4.30) reduces to be the classical one (4.3) when $\overline{\alpha} = 0$. As already pointed out in §4.3.1, the regularisation parameter $\overline{\alpha}$ balances the influence of $\mathcal{D}$ and $\mathcal{R}$. An efficient method to select the optimal $\overline{\alpha}$ will be discussed in the next section. For affine problems where the true solutions require large translations, one may argue that such regularisation might restrict solutions from reaching true solutions. Fortunately our tests will show that this is not the case.

We now express the proposed regularisers in an analytical form in the terms of the six-parameter vector $\mathbf{a} = (a_1, a_2, a_3, a_4, a_5, a_6)^\top \in \mathbb{R}^6$ as follows:

$$\mathcal{R}_1(\mathbf{a}) = \frac{1}{2}\left((1-a_1)^2 + a_2^2 + a_4^2 + (1-a_5)^2\right), \tag{4.32}$$

$$\begin{aligned}\mathcal{R}_2(\mathbf{a}) = \frac{1}{2}\Bigg(&\frac{a_1^2}{3} + \frac{a_2^2}{3} + a_3^2 + \frac{a_4^2}{3} + \frac{a_5^2}{3} + a_6^2 \\ &+ \frac{1}{2}(a_1a_2 + a_4a_5) + a_1a_3 + a_2a_3 + a_4a_6 + a_5a_6 \\ &- \frac{2}{3}(a_1+a_5) - \frac{1}{2}(a_2+a_4) - (a_3+a_6) + \frac{2}{3}\Bigg),\end{aligned} \tag{4.33}$$

$$\begin{aligned}\mathcal{R}_3(\mathbf{a}) = \frac{1}{2}\Bigg(&\frac{4}{3}a_1^2 + \frac{4}{3}a_2^2 + a_3^2 + \frac{4}{3}a_4^2 + \frac{4}{3}a_5^2 + a_6^2 \\ &+ \frac{1}{2}(a_1a_2 + a_4a_5) + a_1a_3 + a_2a_3 + a_4a_6 + a_5a_6 \\ &- \frac{8}{3}(a_1+a_5) - \frac{1}{2}(a_2+a_4) - (a_3+a_6) + \frac{8}{3}\Bigg),\end{aligned} \tag{4.34}$$

$$\mathcal{R}_4(\mathbf{a}) = \frac{1}{2}\left(a_1^2 + a_2^2 + a_3^2 + a_4^2 + a_5^2 + a_6^2\right). \tag{4.35}$$

Further apply the GN approach to solve the discrete minimisation problem:

$$\min_{\mathbf{a}\in\mathbb{R}^6}\{\mathcal{J}_\alpha(\mathbf{a}) = \mathcal{D}(\mathbf{a}) + \overline{\alpha}N\mathcal{R}(\mathbf{a}) = \mathcal{D}(\mathbf{a}) + \alpha\mathcal{R}(\mathbf{a})\}, \tag{4.36}$$

where the factor $N = n^2$ for a square image $n \times n$ is now needed in a multi-resolution setting with $\alpha = \overline{\alpha}N$, since the discrete data term $\mathcal{D}$ as in (4.10) does not contain step-lengths information. The GN perturbation $\delta\mathbf{a}^{(k)}$ for (4.36) is then given by

$$\widetilde{\mathbf{H}}_{\mathcal{J}_\alpha}(\mathbf{a}^{(k)})\delta\mathbf{a}^{(k)} = -\mathbf{g}_{\mathcal{J}_\alpha}(\mathbf{a}^{(k)}) \tag{4.37}$$

where

$$\widetilde{\mathbf{H}}_{\mathcal{J}_\alpha}(\mathbf{a}^{(k)}) = \mathbf{J}^\top(\mathbf{a}^{(k)})\mathbf{J}(\mathbf{a}^{(k)}) + \alpha\mathbf{H}_{\mathcal{R}}(\mathbf{a}^{(k)}) \tag{4.38}$$

and

$$\mathbf{g}_{\mathcal{J}_\alpha}(\mathbf{a}^{(k)}) = \mathbf{g}(\mathbf{a}^{(k)}) + \alpha\nabla_{\mathbf{a}}\mathcal{R}[\mathbf{a}^{(k)}] \tag{4.39}$$

are the approximated Hessian and the gradient of $\mathcal{J}_\alpha$ at $\mathbf{a}^{(k)}$, and $\nabla_{\mathbf{a}}\mathcal{R}[\mathbf{a}^{(k)}]$ and $\mathbf{H}_{\mathcal{R}}(\mathbf{a}^{(k)})$ are respectively the gradient and the Hessian of $\mathcal{R}$ at $\mathbf{a}^{(k)}$. Note that for $\mathcal{R} = \mathcal{R}_1$ we may approximate $\mathbf{H}_{\mathcal{R}_1}\left(\mathbf{a}^{(k)}\right)$ by $\mathbf{I}$ because it helps $\widetilde{\mathbf{H}}_{\mathcal{J}_\alpha}\left(\mathbf{a}^{(k)}\right)$ to be a symmetric positive definite matrix. As before, once we have the GN update $\delta\mathbf{a}^{(k)}$, we can also apply the line-search idea: $\mathbf{a}^{(k+1)} = \mathbf{a}^{(k)} + \zeta^{(k)}\delta\mathbf{a}^{(k)}$.

A connection between our RAR method and the LM method from §4.2.3 can be explained as follows. Consider the regulariser $\mathcal{R}_4$ with a fixed α. Our RAR method defines the perturbation given by

$$[\mathbf{J}^\top(\mathbf{a}^{(k)})\mathbf{J}(\mathbf{a}^{(k)}) + \alpha\mathbf{I}]\delta\mathbf{a}^{(k)} = -[\mathbf{g}(\mathbf{a}^{(k)}) + \alpha\mathbf{I}\mathbf{a}^{(k)}], \tag{4.40}$$

which is a solution of the following minimisation problem:

$$\min_{\delta\mathbf{a}\in\mathbb{R}^6}\frac{1}{2}||F(\mathbf{a}^{(k)}) + \mathbf{J}\left(\mathbf{a}^{(k)}\right)\delta\mathbf{a}^{(k)} - \mathbf{R}||_2^2 + \frac{\alpha}{2}||\mathbf{a}^{(k)} + \delta\mathbf{a}^{(k)}||_2^2. \tag{4.41}$$

If the second term in (4.41) is replaced by $\frac{\alpha}{2}||\delta\mathbf{a}^{(k)}||_2^2$, i.e. we set $\mathbf{a}^{(k)} = \mathbf{0}$, we recover the old LM perturbation (4.18):

$$[\mathbf{J}^\top(\mathbf{a}^{(k)})\mathbf{J}(\mathbf{a}^{(k)}) + \alpha\mathbf{I}]\delta\mathbf{a}^{(k)} = -g(\mathbf{a}^{(k)}), \tag{4.42}$$

which is a solution of the minimisation problem:

$$\min_{\delta\mathbf{a}\in\mathbb{R}^6}\frac{1}{2}||F(\mathbf{a}^{(k)}) + \mathbf{J}\left(\mathbf{a}^{(k)}\right)\delta\mathbf{a}^{(k)} - \mathbf{R}||_2^2 + \frac{\alpha}{2}||\delta\mathbf{a}^{(k)}||_2^2. \tag{4.43}$$

Although the second term in (4.43) can be viewed as a regulariser, a Tikhonov-like term, for the perturbation $\delta\mathbf{a}$, the main problem with using this latter type of regularisers is that we cannot directly control the characteristics of the solution. In other words, this approach does not take account into *a priori* information about the characteristics of solutions, which is the main task of regularisation. In contrast, our RAR approach regularises the current step $\left(\mathbf{a}^{(k)} + \delta\mathbf{a}\right)$ and so does control the characteristics of the solution.

4.6 A cooling method for the RAR parameter

As will be shown in §4.7, our RAR model is more robust than the standard affine model due to being less demanding on good initial guesses. However the standard model does not need a

regularisation parameter α. Here an algorithm to select the optimal α on the finest level L is presented first, followed by the idea to a multi-resolution setting to minimise the extra work.

There are many ways to select α — one option is to use the 'cooling' process (i.e. continuation) in an adaptive manner (see Haber and Oldenberg [66], Newman and Hoversten [108], Chen *et al* [29], and Lelièvre and Oldenberg [92]). The basic idea is to start with a high initial value of α and then slowly reduce α in such a way that the solution obtained using it is an excellent starting point for the next minimisation problem, in order to decrease $\mathcal{J}_\alpha$.

The initial α_1 is first estimated so that $\alpha_1 \mathbf{H}_{\mathcal{R}}(\mathbf{a}_1^{(0)})$ dominates the $\mathbf{J}^\top(\mathbf{a}_1^{(0)})\mathbf{J}(\mathbf{a}_1^{(0)})$ component in (4.38), where $\mathbf{a}_1^{(0)}$ is the initial guess solution. At the $(l+1)$th step we set

$$\alpha_{l+1} = \eta \alpha_l \in [\alpha_0, \alpha_1], \tag{4.44}$$

where η is a constant, usually chosen to be about 0.5, and α_0 is a small positive number, e.g. 5×10^{-5}. Subsequently, we apply α_{l+1} and the initial guess solution obtained by the previous iteration $\mathbf{a}_{l+1}^{(0)} = \mathbf{a}_l$ with the associated inner loop to obtain the minimum $\mathbf{a}_{l+1}$ within some tolerance. As mentioned in [66], since the functional $\mathcal{J}_\alpha$ changes at each outer loop iteration, the demand of decreasing the value of the same functional is not reasonable we impose the so-called *consistent condition* to ensure that the solution $\mathbf{a}_{l+1}$ and parameter α_{l+1} are acceptable:

$$\mathcal{J}_{\alpha_{l+1}}(\mathbf{a}_{l+1}) = \mathcal{D}(\mathbf{a}_{l+1}) + \alpha_{l+1}\mathcal{R}(\mathbf{a}_{l+1}) < \mathcal{J}_{\alpha_{l+1}}(\mathbf{a}_l) = \mathcal{D}(\mathbf{a}_l) + \alpha_{l+1}\mathcal{R}(\mathbf{a}_l). \tag{4.45}$$

If this condition is not satisfied, we increase η (usually to 0.9) and re-start the step. Our experience suggests that the criterion given by

$$\frac{\|\mathbf{a}_{l+1} - \mathbf{a}_l\|}{\max\{\|\mathbf{a}_{l+1}\|, \|\mathbf{a}_l\|\}} < \delta \tag{4.46}$$

is suitable, where $\delta > 0$ is small (normally set to 5×10^{-4}). The process of solving the problem (4.36) for $\mathbf{a}$ with a given α (by the new RAR solver) will be denoted by

$$\mathbf{a}_\ell \leftarrow \mathit{Solver\_RAR}(\mathbf{T}_\ell, \mathbf{R}_\ell, \mathbf{a}_\ell, \alpha, \mathit{tol})$$

for tolerance *tol* and the maximum number of iterations IMAX.

Finally, we summarise this unilevel cooling process as follows:

Algorithm 4.6.1 (Registration through cooling)

$[\mathbf{a}^*, \alpha^*] \leftarrow \mathit{cooling}(\mathbf{T}, \mathbf{R}, \mathbf{a}^{(0)}, \alpha^{(0)})$

- Set $l = 1$, $\eta = 0.5$, $\mathbf{a}_l = \mathbf{a}^{(0)}$, $\alpha_0 = 5 \times 10^{-5}$ and $\alpha_1 = \alpha^{(0)}$. Set IMAX= 25 and $\mathit{tol} = 10^{-3}$.
- Outer iteration: For $l = 1, 2, 3, ...$

  –1. Set $\alpha_{l+1} = \eta \alpha_l$ in $[\alpha_0, \alpha_l]$

  –2. Inner iteration: $\mathbf{a}_{new} \leftarrow \mathit{Solver\_RAR}(\mathbf{T}, \mathbf{R}, \mathbf{a}_l, \alpha_{l+1}, \mathit{tol})$.

  –3. If $\mathcal{J}_{\alpha_{l+1}}(\mathbf{a}_{new}) < \mathcal{J}_{\alpha_{l+1}}(\mathbf{a}_l)$

–3.1. Set $\mathbf{a}_{l+1} = \mathbf{a}_{new}$, $\eta = 0.5$, $l = l + 1$, and go to 4

Else

–3.2. Set $\eta = 0.9$, and go to 4

–4. Check for convergence using the criterion (4.46).

If not satisfied, then return to 1, else, exit to the next step to stop.

- Set $\mathbf{a}^* = \mathbf{a}_{new}$ and $\alpha^* = \alpha_l$.

In the above algorithm, one notes that each minimisation may not to be solved exactly within IMAX iterations. Even so, the algorithm can be expensive for large images due to accumulated cost. Then our first robust algorithm will be the following.

Algorithm 4.6.2 (The basic RAR method)

1. *Input tol, given images* $\mathbf{T}$, $\mathbf{R}$. *Set* $\alpha = 1$ *(optional).*

2. *Obtain the optimal regularisation parameter* α *(through cooling) via Algorithm 4.6.1:*

$$[\mathbf{a}^{(0)}, \alpha] \leftarrow cooling(\mathbf{T}, \mathbf{R}, \mathbf{a}, \alpha).$$

3. *Solve the RAR problem (4.36) on the finest level:*

$$\mathbf{a} \leftarrow Solver\_RAR(\mathbf{T}, \mathbf{R}, \mathbf{a}, \alpha, tol).$$

In order to save computational work, we propose to use a hierarchy of L grids (with level L the finest and level 1 the coarsest one) as in §4.4.4. The optimal α is searched only on the coarsest level 1, followed by the idea of §4.4.4 to provide finer level initial guesses. The whole procedure is summarised in Algorithm 4.6.3.

Algorithm 4.6.3 (Multilevel strategy for optimal α and reliable initial solution)

$[\mathbf{a}_\ell, \alpha_\ell] \leftarrow RAR\_multiresolution(\mathbf{T}_\ell, \mathbf{R}_\ell, \mathbf{a}_\ell, \alpha_\ell, \ell, tol)$

- If $\ell = 1$
  - $\mathbf{a}_\ell = (1, 0, 0, 0, 1, 0)^\top$ or use Methods $1 - 3$ in §4.4 to work out an initial $\mathbf{a}_\ell$
  - $\alpha_\ell = C$ $\quad [C > 0$ should be large enough e.g. $C = 1000]$
  - $[\mathbf{a}_\ell, \alpha_\ell] \leftarrow cooling(\mathbf{T}_\ell, \mathbf{R}_\ell, \mathbf{a}_\ell, \alpha_\ell)$
- Else
  - $\mathbf{T}_{\ell-1} = coarsen\,(\mathbf{T}_\ell)\,, \quad \mathbf{R}_{\ell-1} = coarsen\,(\mathbf{R}_\ell)$
  - $[\mathbf{a}_{\ell-1}, \alpha_{\ell-1}] \leftarrow RAR\_multiresolution(\mathbf{T}_{\ell-1}, \mathbf{R}_{\ell-1}, \mathbf{a}_{\ell-1}, \alpha_{\ell-1}, \ell - 1, tol)$

- $\mathbf{a}_\ell \leftarrow interpolate(\mathbf{a}_{\ell-1})$ as follows:
  $a_{i_\ell} = a_{i_{\ell-1}}$ for $i = 1, 2, 4, 5$ (the elements of the affine transformation matrix) and
  $a_{i_\ell} = 2a_{i_{\ell-1}}$ for $i = 3, 6$ (the elements of the translation matrix).
- $\alpha_\ell = 4\alpha_{\ell-1}$ [Recall that $\alpha_\ell = \overline{\alpha} n_l^2$ and $n_l = 2n_{l-1}$]
- $\mathbf{a}_\ell \leftarrow Solver\_RAR(\mathbf{T}_\ell, \mathbf{R}_\ell, \mathbf{a}_\ell, \alpha_\ell, tol, \text{IMAX})$

Algorithm 4.6.4 (The refined RAR method)

1. *Input tol and set* $\mathbf{T}_L = \mathbf{T}$, $\mathbf{R}_L = \mathbf{R}$ *on the finest level. Set toln,* $MAXN$.

2. *Obtain the optimal regularisation parameter* α *(on the coarsest level* 1 *through cooling) and a good initial solution (through multi-resolution) via Algorithm 4.6.3:*

$$[\mathbf{a}^{(0)}, \alpha] \leftarrow RAR\_multiresolution(\mathbf{T}_L, \mathbf{R}_L, \mathbf{a}_L, \alpha_L, L).$$

3. *Solve the RAR problem (4.36) on the finest level* $l = L$ *using the found* α*:*

$$\mathbf{a}_\ell \leftarrow Solver\_RAR(\mathbf{T}_\ell, \mathbf{R}_\ell, \mathbf{a}_\ell, \alpha_\ell, toln, MAXN).$$

4.7 Numerical experiments

In this section, some results to illustrate the proposed algorithms are presented. The first example (Example 4.7.1) is used to defend the integrity of the RAR method, i.e. problems that possess genuinely large components in $\mathbf{a}$ are not penalised by the proposed method (the regularisation). The second example (Example 4.7.2) is employed to shown that, for a nontrivial affine problem, the standard affine model even when Method 4 (§4.4.4) can fail to register properly while the RAR models (especially Algorithm 4.6.4) can register successfully. The final example (Example 4.7.3)aims to show that, for the deformable problem (Example 4.4.1), the RAR method can provide a better initial solution than Method 4 (§4.4.4) which leads to even fewer number of FAS cycles by a deformable method (§4.3).

Example 4.7.1 *We consider a pair of synthetic images as in Figure* 4.6 $(a) - (b)$ *with the images of size* 512×512. *Clearly one expects* $\mathbf{a}$ *will require large values.*

Using Algorithm 4.6.4 *with* $\mathcal{R}_1$, *we find that* $\mathbf{a}_{\mathcal{R}_1}$=(0.2561, 0.4800, −134.4109, −0.2399, 0.8000, 275.9836)$^\top$ *which is evidently not penalised by regularisation. Similar solutions are obtained by* $\mathcal{R}_2, \mathcal{R}_3, \mathcal{R}_4$. *The successfully registered images using these* 4 *regularisers are respectively shown in Figure* 4.6 (c), (d), (e) *and* (f). *Here* $\mathbf{a}_{\mathcal{R}_1}$ *is the solution obtained from the regulariser* $\mathcal{R}_1$.

Example 4.7.2 *We consider an affine registration problem for a pair of MR images of a human head as in Figure* 4.7 $(a) - (b)$, *where* $n_1 = n_2 = 256$. *We compare the GN and LM methods with Method* 4 *(Algorithm* 4.4.1*) with our RAR method (Algorithm* 4.6.4*).*

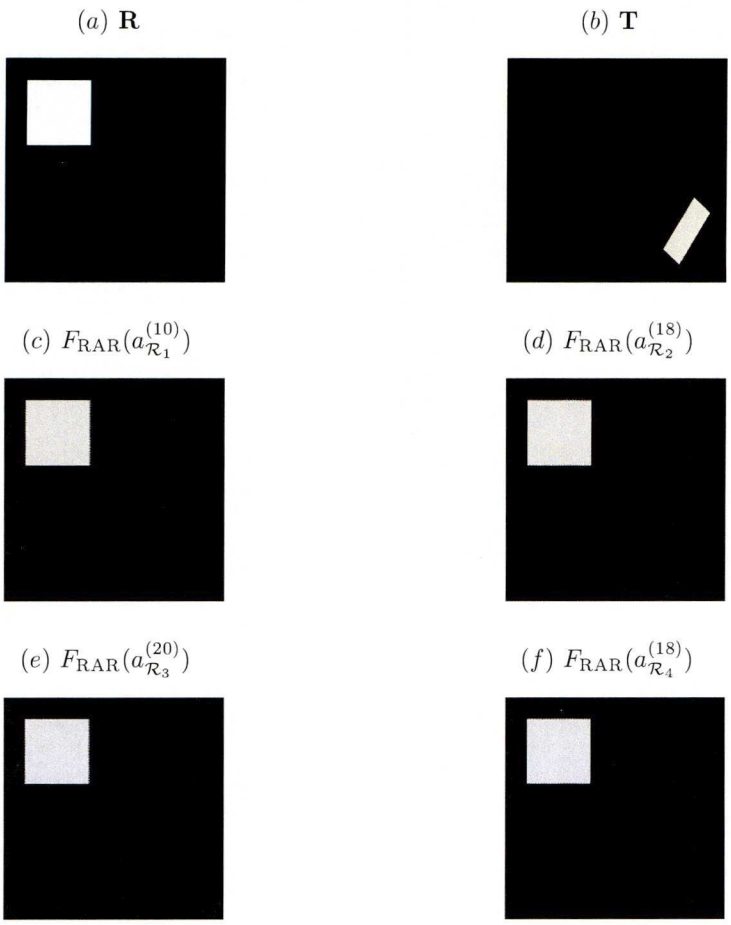


Figure 4.6: Example 4.7.1: Correct registration results (requiring large affine parameters) of a pair of synthetic images by our RAR model. The first row shows the reference (a) **R** and (b) the template **T**. The second and third rows show the registered images $(c)-(f)$ from our 4 regularisers $\mathcal{R}_1-\mathcal{R}_4$, respectively.

Since $\max\{||\mathbf{a}^*-\mathbf{a}_{\mathcal{R}_i}||_2/||\mathbf{a}^*||_2|i=1,2,3,4\}=0.0069$, *this means that our method converges to the true solution. Moreover, the registered images obtained from* 4 *different regularisers shown in Figure* 4.8 $(a)-(d)$ *are almost identical. Comparing those results obtained from the GN and LM methods (see Figure* 4.7 $(c)-(d)$*) and our RAR method (see Figure 4.8* $(a)-(d)$*), one notes that the proposed latter method is more robust than the former methods.*

Example 4.7.3 *Finally, we re-solve Example* 4.4.1 *to show that Algorithm* 4.6.4 *is better than Algorithm* 4.4.1 *in affine pre-registration for the purpose of using a deformable model (via FAS algorithm).*

Here we show in Figure 4.9 $(a)-(d)$ *the four respective pre-registration images from our* 4 *regularisers, and they appear identical. Indeed, using any of them to start FAS (*§4.3.3*) gives the same result as shown in Figure* 4.9 $(e)-(f)$ *using* $(a)-(b)$ *respectively. Moreover the details*

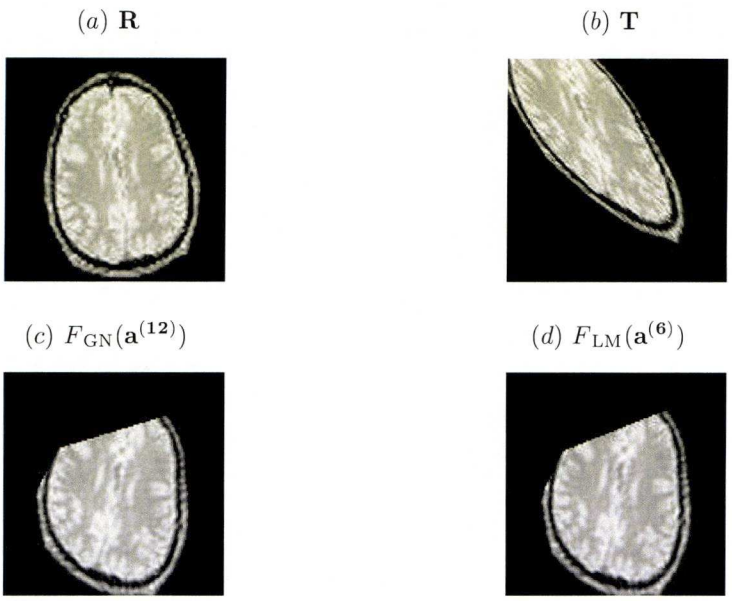


Figure 4.7: Example 4.7.2: Failed registration results of GN and LM with Method 4 (Algorithm 4.4.1). The first row shows the reference image: (*a*) **R** and the template image: (*b*) **T**. The second row presents the registered images: (*c*) $F_{\mathrm{GN}}(\mathbf{a}^{(12)})$ and (*d*) $F_{\mathrm{LM}}(\mathbf{a}^{(6)})$.

from Figure 4.9 $(a)-(d)$ *are visually more pleasing than Figure* 4.5 $(e)-(f)$ *(especially at the upper region).*

To show the quantitative gain from using our Algorithm 4.6.4, we now present the comparable results in Table 4.1 for clarity, where 'Out.Iters' (same as l in step 3.1 of Algorithm 4.6.1) is the number of the outer iterations by Algorithm 4.6.2 and 'Avg.Iters' means the average of the number of inner iterations, given by

$$\text{Avg.Iters} = \frac{\text{The number of accumulated iterations by } \textit{Solver\_RAR} \text{ on the finest level}}{\text{The number of updates for parameter } \alpha \text{ (via steps 3.1 and 3.2 in Algorithm 4.6.1)}}$$

Clearly apart form the quality improvement over standard models (as illustrated before), much speed gain can be observed in Table 4.1 with our recommended Algorithm 4.6.4.

To summarise, in these and other tests, we have compared the performance of Algorithm 4.6.2 with 4.6.4. While both give comparable results, Algorithm 4.6.4 is much cheaper due to using a coarse level to work out for α.

4.8 Conclusions

Parametric registration via a nonlinear least-square model offers a fast registration method. However the commonly used iterative methods such as the GN and LM methods often have convergence difficulties, due to lack of good initial solutions, so the resulting nonlinear model is often not robust.

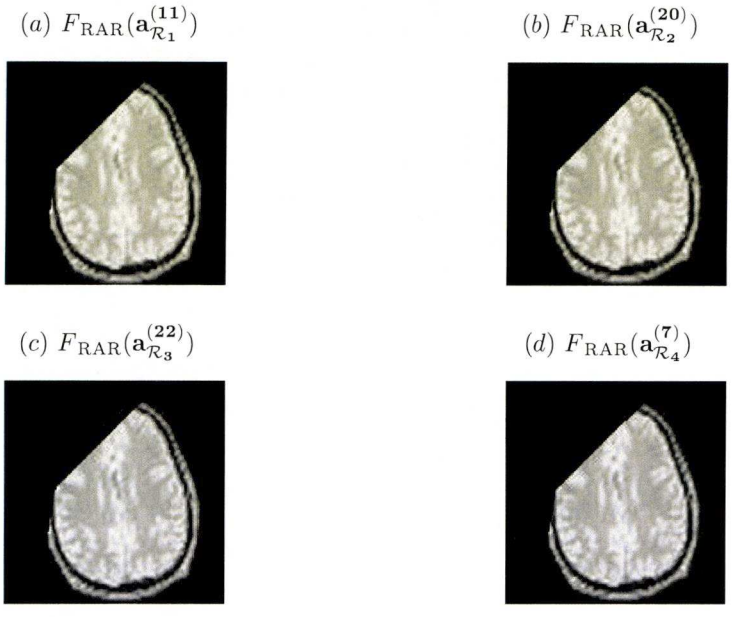


Figure 4.8: Example 4.7.2 re-solved: Correct registration results using our RAR method (Algorithm 4.6.4) with 4 regularisers $\mathcal{R}_1 - \mathcal{R}_4$, respectively shown in $(a) - (d)$.

In this chapter, we first examined the robustness issue of the GN and LM methods for affine image registration problems by reviewing four existing methods for getting good initial guesses. It turns out that there are always difficult cases for which these initial guesses are not sufficient. Such cases include getting pre-registration images for deformable registration problems; we reviewed the diffusion model and used a FAS-NMG method for testing purposes. Second, we introduced a regularised affine registration (RAR) model that is less demanding than the standard model for initial guesses. To find the optimal regularisation parameter in an efficient way, we used a coarse-to-fine approach to initialise the RAR model. Numerical results showed that the developed multilevel algorithm is generally reliable and robust in i) solving the affine image registration problems ii) providing a good initial guess for deformable models.

Recently there was new work introduced by Haber and Modersitzki [63] attempting to combine parametric and non-parametric models and we believe our idea of regularising the parametric coefficients should be applicable there as well.

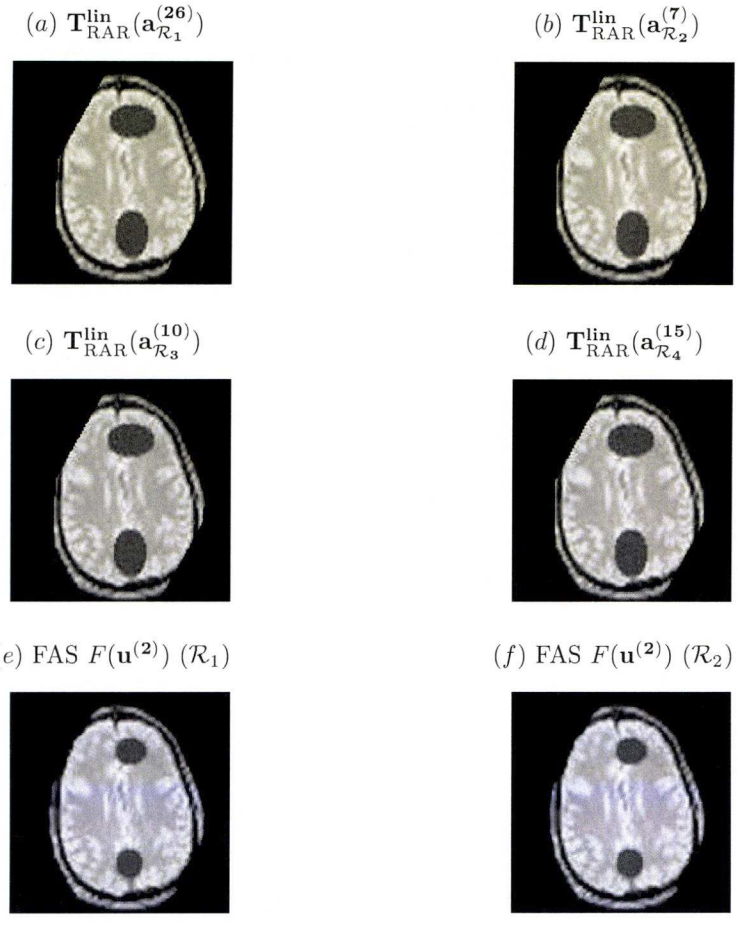


Figure 4.9: Example 4.4.1 re-solved and improved: The affine pre-registration steps $(a)-(d)$ using Algorithm 4.6.4 with different regularisers $\mathcal{R}_1-\mathcal{R}_4$, respectively. The last row $(e)-(f)$ shows the respective registered images by FAS method with $(a)-(b)$ as initial solutions (using the $(c)-(d)$ gives almost identical solutions). Clearly less FAS cycles (i.e. 2) are needed than before (i.e. 6).

Example	Image	Algorithm 4.4.1 (GN)	Algorithm 4.6.2 (S_1)	Algorithm 4.6.4 (S_1)
Number	size N	Iters/Ini.Cpu/Cpu	Out.Iters/Avg.Iters/Cpu	Iters/Ini.Cpu/Cpu
4.7.1	128^2	8/0.1/0.4	3/9/1.7	9/0.2/0.5
	256^2	7/0.2/1.7	3/9/5.7	7/0.3/1.7
	512^2	9/0.9/9.5	3/10/30.3	10/1.1/9.3
	1024^2	10/3.6/44.9	3/10/125.3	10/3.6/41.2
4.7.2	128^2	7/0.1/0.4	9/12/5.2	11/0.3/0.7
	256^2	12/0.2/3.5	9/12/51.7	11/0.3/2.6
	512^2	8/1.1/10.3	9/11/251.5	10/2.0/11.7
	1024^2	24/4.8/139.5	9/11/810.1	12/8.2/58.0

Table 4.1: Comparison of Algorithm 4.4.1, 4.6.2, 4.6.4 using Examples 4.7.1–4.7.2 with varying N.

Chapter 5

A Robust Multigrid Approach for Variational Image Registration Models

Variational registration models are non-rigid and deformable imaging techniques for accurate registration of two images. As with other models for inverse problems using Tikhonov regularisation, they must have a suitably chosen regularisation term as well as a data fitting term. One distinct feature of registration models is that their fitting term is always highly nonlinear and this nonlinearity restricts the class of numerical methods that are applicable. This chapter first reviews the current state of art numerical methods for such models and observes that the nonlinear fitting term is mostly 'avoided' in developing fast multigrid methods. It then proposes a unified approach for designing fixed-point type smoothers for multigrid methods. The diffusion registration model (second order equations) and a curvature model (fourth order equations) are used to illustrate our robust methodology. Analysis of the proposed smoothers and comparisons to other methods are given. As expected of a multigrid method, being many orders of magnitude faster than the unilevel gradient descent approach, the proposed numerical approach delivers fast and accurate results for a range of synthetic and real test images.

5.1 Introduction

Given a reference image R and a template image T, the image registration problem can be posed as a minimisation problem of the *joint* energy functional given by

$$\min_{\boldsymbol{u}}\{\mathcal{J}_\alpha(\boldsymbol{u}) = \mathcal{D}^{\mathrm{SSD}}(\boldsymbol{u}) + \alpha\mathcal{R}(\boldsymbol{u})\}. \tag{5.1}$$

Here the image intensities of R and T are assumed to be comparable and we adopt the SSD functional

$$\mathcal{D}^{\mathrm{SSD}}(\boldsymbol{u}) = \tfrac{1}{2}\int_\Omega (T(\mathbf{x}+\boldsymbol{u}(\mathbf{x})) - R(\mathbf{x}))^2\, d\mathbf{x}, \tag{5.2}$$

to quantify distance or similarity of two given images R and T. Recall that R and T are modelled as the continuous functions mapping from an image domain $\Omega \subset \mathbb{R}^2$ into $V \subset \mathbb{R}_0^+$

and the components u_1 and u_2 of the deformation field $\boldsymbol{u} : \mathbb{R}^2 \to \mathbb{R}^2$ are the functions of the variable $\mathbf{x} = (x_1, x_2)^\top$ in the image domain Ω. Without loss of generality we assume that $\Omega = [0,1]^2 \subset \mathbb{R}^2$ and $V = [0,1]$ for 2D gray-scale images.

In this chapter our main concern is to address the fast and effective solutions of the resulting PDE systems from (5.1). We consider two regularisers: firstly the *diffusion regulariser* as introduced by Fischer and Modersitzki [46]:

$$\mathcal{R}^{\text{diff}}(\boldsymbol{u}) = \tfrac{1}{2}\sum_{l=1}^{2}\int_{\Omega} |\nabla u_l(\mathbf{x})|^2 \, d\mathbf{x} \tag{5.3}$$

and secondly the *curvature regulariser* as introduced by Fischer and Modersitzki [47]:

$$\mathcal{R}^{\text{FMcurv}}(\boldsymbol{u}) = \tfrac{1}{2}\sum_{l=1}^{2}\int_{\Omega} (\Delta u_l)^2 d\mathbf{x} \tag{5.4}$$

as respective examples of second order PDEs and fourth order PDEs; our method will also be applicable to other models that lead to second or fourth order PDEs, e.g. the elastic model [104], the total variation model [51, 53], the modified total variation model (see Chapter 6 later), and other curvature models [35, 79, 78, 73, 75].

Although the multigrid techniques have been successfully used for numerical solutions for deformable image registration [53, 54, 65, 76, 73, 83, 89, 131, 145], none of the existing variants are optimal implementations. The nonlinear fitting term is mostly avoided in these works. We remark that other imaging models [27, 24, 31, 118] do not have such problems due to the fitting term; see §3.6 for a brief review of existing multigrid methods for deformable image registration.

The rest of the chapter is organized as follows. In §5.2, we consider our first model problem of diffusion registration, surveying and discussing its numerical treatments. A new fixed-point smoother is proposed and analysed in §5.3, for the FAS-NMG approach for the underlying nonlinear Euler-Lagrange systems. In §5.4,, we consider our second model problem of curvature registration and demonstrate how to use our proposed method. Experimental results from medical test images are illustrated in §5.5, in order to show the excellent performance of the proposed numerical scheme compared with other methods with conclusions summarised in §5.6.

5.2 The diffusion registration model and its numerical methods

We now introduce our first model and review briefly various solution methods paying particular attention to robustness of multigrid methods. The model itself is not particularly more important than other models from [104] but we use it to illustrate the fast solution issues of image registration. Below we use $\partial_{x_l} F = \frac{\partial F}{\partial x_l}$ and $\partial_{x_1 x_2} F = \frac{\partial F}{\partial x_1 \partial x_2}$.

5.2.1 The diffusion model

The minimiser $\boldsymbol{u} = (u_1(\mathbf{x}), u_2(\mathbf{x}))^\top$ of the energy functional $\mathcal{J}_\alpha$ in (5.1), defined by (5.2) and (5.3) satisfies the Euler-Lagrange equation [104], given by the following system of two coupled,

nonlinear, and elliptic partial differential equations (PDEs):

$$\begin{cases} \mathcal{N}_1(\boldsymbol{u}) = -\alpha\Delta u_1 + \overbrace{(T_{\boldsymbol{u}} - R)\,\partial_{u_1} T_{\boldsymbol{u}}}^{f_1(\boldsymbol{u})} = 0, \\ \mathcal{N}_2(\boldsymbol{u}) = \underbrace{-\alpha\Delta u_2 + \overbrace{(T_{\boldsymbol{u}} - R)\,\partial_{u_2} T_{\boldsymbol{u}}}^{f_2(\boldsymbol{u})}}_{} = 0, \end{cases} \tag{5.5}$$

subject to the homogenous Neumann's boundary conditions

$$\partial_{\boldsymbol{n}} u_1 = \partial_{\boldsymbol{n}} u_2 = 0 \text{ on } \partial\Omega. \tag{5.6}$$

Here the nonlinear functions $f_1(\boldsymbol{u})$, $f_2(\boldsymbol{u})$ are from the fitting term $\mathcal{D}^{\text{SSD}}$, which is nonlinear (as remarked) and a key feature of registration models distinct from other imaging models [27]. Refer to [46, 47, 48, 49, 51, 53, 54, 62, 65, 59, 60, 61, 76, 72, 79, 78, 73, 75, 83, 89, 90, 94, 104, 131, 145]. In fact, the nonlinear coupling of the two PDEs is through the term $T(\boldsymbol{u})$. Hereby, Δ denotes the Laplace operator, and $\boldsymbol{n} = (n_1, n_2)^\top$ is the outward unit vector normal to the image boundary $\partial\Omega$. Note that the first and second terms in (5.5) are the first variations of the regulariser term $\mathcal{R}$ and the data term $\mathcal{D}^{\text{SSD}}$, respectively.

5.2.2 Discretisation by a finite difference method

For simplicity, let $\left(u_l^h\right)_{i,j} = u_l^h\left(x_{1_i}, x_{2_j}\right)$ denote the grid function for $l = 1, 2$ with grid spacing $h = (h_1, h_2) = (1/n_1, 1/n_2)$. Applying finite difference schemes based on the cell-centered grid points to discretise (5.5), the discrete Euler-Lagrange equations at a grid point (i, j) over the discrete domain,

$$\Omega_h = \{\mathbf{x} \in \Omega | \mathbf{x} = (x_{1_i}, x_{2_j})^\top = ((2i-1)\,h_1/2, (2j-1)\,h_2/2)^\top,\ 1 \le i \le n_1, \quad 1 \le j \le n_2\} \tag{5.7}$$

are given by

$$\begin{cases} \mathcal{N}_1^h(\boldsymbol{u}^h)_{i,j} = -\alpha\mathcal{L}^h(u_1^h)_{i,j} + f_1^h(u_1^h, u_2^h)_{i,j} = g_{1_{i,j}}^h \\ \mathcal{N}_2^h(\boldsymbol{u}^h)_{i,j} = -\alpha\mathcal{L}^h(u_2^h)_{i,j} + f_2^h(u_1^h, u_2^h)_{i,j} = g_{2_{i,j}}^h \end{cases} \tag{5.8}$$

with the following notation

$$\begin{aligned} -\mathcal{L}^h(u_l^h)_{i,j} &= (1/h_1^2)((\Sigma)_{i,j}\,(u_l^h)_{i,j} - \left(\overline{\Sigma}\right)_{i,j}(u_l^h)_{i,j}), \\ (\Sigma)_{i,j} &= 2(1+\gamma^2), \quad \gamma = h_1/h_2, \\ \left(\overline{\Sigma}\right)_{i,j}(u_l^h)_{i,j}) &= (u_l^h)_{i+1,j} + (u_l^h)_{i-1,j} + \gamma^2(u_l^h)_{i,j+1} + \gamma^2(u_l^h)_{i,j-1} \\ f_1^h\left(u_1^h, u_2^h\right)_{i,j} &= (T_{i,j}^{h^*} - R_{i,j}^h)((T_{i+1,j}^{h^*} - T_{i-1,j}^{h^*})/\,(2h_1)), \\ f_2^h\left(u_1^h, u_2^h\right)_{i,j} &= (T_{i,j}^{h^*} - R_{i,j}^h)((T_{i,j+1}^{h^*} - T_{i,j-1}^{h^*})/\,(2h_2)), \\ T_{i,j}^{h^*} &= T^h(i + (u_1^h)_{i,j}, j + (u_2^h)_{i,j}), \\ (\boldsymbol{u}^h)_{i,j} &= ((u_1^h)_{i,j}, (u_2^h)_{i,j})^\top. \end{aligned}$$

Here $g_{1_{i,j}}^h = g_{2_{i,j}}^h = 0$ on the finest grid in multigrid setting to be used shortly. We note that the approximations in (5.8) need to be adjusted at the image boundary $\partial\Omega_h$ using the homogeneous

Neumann boundary conditions

$$(u_l^h)_{i,1} = (u_l^h)_{i,2}, \quad (u_l^h)_{i,n_2} = (u_l^h)_{i,n_2-1}, \quad (u_l^h)_{1,j} = (u_l^h)_{2,j}, \quad (u_l^h)_{n_1,j} = (u_l^h)_{n_1-1,j}, \quad (5.9)$$

5.2.3 Review of non-multigrid numerical solvers

The **first** commonly used method is a gradient descent approach solving, instead of the nonlinear elliptic system (5.5), the nonlinear parabolic system

$$\begin{cases} \partial_t u_1 - \alpha\Delta u_1 = -\left(I_{\boldsymbol{u}}^{'} - R\right)\partial_{u_1} I_{\boldsymbol{u}}^{'} \\ \partial_t u_2 - \alpha\Delta u_2 = -\left(I_{\boldsymbol{u}}^{'} - R\right)\partial_{u_2} I_{\boldsymbol{u}}^{'} \end{cases}, \quad (5.10)$$

where $\boldsymbol{u} = \boldsymbol{u}(\mathbf{x}, t) = (u_1(\mathbf{x}, t), u_2(\mathbf{x}, t))^\top$ will converge to the solution of (5.5) when $t \to \infty$, with the initial solution $\boldsymbol{u}(\mathbf{x}, 0)$, typically $\boldsymbol{u}(\mathbf{x}, 0) = 0$. The advantage is that various time-marching schemes can be used to solve (5.10) in order to circumvent the nonlinearity on the right-hand side. For example, the semi-implicit scheme can be proceeded as follows (in matrix vector form obtained from the discretised version of equation (5.10) by §5.2.2)

$$\begin{cases} u_1^{(k+1)} = (\mathbf{I} - \alpha\tau\sum_{l=1}^{2}\mathbf{A}_l)^{-1}(u_1^{(k)} - \tau f_1(u_1^{(k)}, u_2^{(k)})) \\ u_2^{(k+1)} = (\mathbf{I} - \alpha\tau\sum_{l=1}^{2}\mathbf{A}_l)^{-1}(u_2^{(k)} - \tau f_2(u_1^{(k)}, u_2^{(k)})) \end{cases}. \quad (5.11)$$

Here, $\mathbf{I}$ is the identity matrix, $f_l\left(u_1^k, u_2^k\right)$ is the discretised version of the second term in (5.5), $\tau > 0$ is the time-step determined by a forward difference approximation of the time derivative $\partial_t u_l$, and $\mathbf{A}_l$ is the coefficient matrix from discretisation of the Laplace operator Δ along the l-coordinate direction subject to Neumann boundary conditions. We note that the DCT-based method in [104] and the FT-based method in [91, 135] are optional to exactly solve the linear system (5.11).

The **second** method, an additive operator splitting (AOS) scheme in [46, 93, 138], is faster and more efficient than the standard semi-implicit scheme (5.11). The basic idea is to replace the inverse of the sum by a sum of inverses. The corresponding iterations are then defined by

$$\begin{cases} u_1^{(k+1)} = \frac{1}{2}\sum_{l=1}^{2}\left(\mathbf{I} - 2\alpha\tau\mathbf{A}_l\right)^{-1}\left(u_1^{(k)} - \tau f_1(u_1^{(k)}, u_2^{(k)})\right) \\ u_2^{(k+1)} = \frac{1}{2}\sum_{l=1}^{2}\left(\mathbf{I} - 2\alpha\tau\mathbf{A}_l\right)^{-1}\left(u_2^{(k)} - \tau f_2(u_1^{(k)}, u_2^{(k)})\right) \end{cases}, \quad (5.12)$$

which is much cheaper than those obtained from (5.11) because the two tri-diagonal systems in each component are solved per iteration rather than the 5-band system. We remark that the AOS scheme is of $\mathcal{O}(N)$ ($N = n_1 n_2$), while the DCT-based method is of $\mathcal{O}(N \log N)$.

The **third** method, the one-cycle multi-resolution (or a coarse-to-fine FMG technique), is proposed by Lu et al. [94]. The idea is to solve (5.5) first with the Newton-Gauss-Seidel relaxation method given by

$$\begin{cases} u_1^{(new)} = u_1^{(old)} - \frac{\mathcal{N}_1(\boldsymbol{u}^{(old)})}{\partial_{u_1}\mathcal{N}_1(\boldsymbol{u}^{(old)})} \\ u_2^{(new)} = u_2^{(old)} - \frac{\mathcal{N}_2(\boldsymbol{u}^{(old)})}{\partial_{u_2}\mathcal{N}_2(\boldsymbol{u}^{(old)})} \end{cases} \quad (5.13)$$

on the coarsest (lowest) resolution, and then interpolate the coarse solution with an appropriate method as a good initial guess solution to the next finer resolution. This process is repeated until (5.5) has been solved by (5.13) on the finest (highest) or original resolution. Although, this method provides well matched images in most of the cases and the basic idea is essentially the same as a FMG method as in [12, 67, 134, 139, 140], one should note that convergence cannot be proved even for much simpler equations [134]. This is due to the fact that the solution on the fine level depends strongly on the coarse one. As a consequence, errors introducing from the interpolation procedure may propagate in such a way that they spoil the overall results completely.

Although the above three methods are easy to implement on a unilevel and multilevel, they are not as efficient as a multigrid method.

5.2.4 Review of multigrid solvers and previous work

Multigrid techniques are widely used as fast methods for various PDEs [12, 67, 134, 139, 140]. For deformable image registration models, we give a brief review here [53, 54, 65, 76, 73, 83, 89, 131, 145].

This section first reviews the basic FAS-NMG algorithm before discussing previous work on solving (5.5). As it turns out, two approaches considered are both more efficient than a unilevel method but none are robust solvers.

The full approximation scheme (FAS)

For a nonlinear problem, the use of a full approximation scheme (the nonlinear multigrid method (NMG) by Brandt [11]) is natural. The FAS technique has been tried in various image processing applications; see e.g. [6, 7, 13, 18, 22, 24, 31, 52, 53, 122, 123]. We now denote our system of two nonlinear PDEs (5.5) by

$$\begin{cases} \mathcal{N}_1^h\left(\boldsymbol{u}^h\right) = g_1^h, \\ \mathcal{N}_2^h\left(\boldsymbol{u}^h\right) = g_2^h, \end{cases} \tag{5.14}$$

separating the linear operator $\mathcal{L}^h$ from the nonlinear operator f_l^h $(l = 1, 2)$ on a general fine grid with step size $h = (h_1, h_2)$ on Ω_h.

Let $\boldsymbol{v}^h = \left(v_1^h, v_2^h\right)^\top$ be the result computed by performing a few steps with a smoother (**pre-smoothing step**) on the fine-grid problem. Then, the algebraic error $\boldsymbol{e}^h$ of the solution is given by $\boldsymbol{e}^h = \boldsymbol{u}^h - \boldsymbol{v}^h$. The residual equation system is given by

$$\begin{cases} \mathcal{N}_1^h\left(\boldsymbol{v}^h + \boldsymbol{e}^h\right) - \mathcal{N}_1^h\left(\boldsymbol{v}^h\right) = g_1^h - \mathcal{N}_1^h\left(\boldsymbol{v}^h\right) = r_1^h \\ \mathcal{N}_2^h\left(\boldsymbol{v}^h + \boldsymbol{e}^h\right) - \mathcal{N}_2^h\left(\boldsymbol{v}^h\right) = g_2^h - \mathcal{N}_2^h\left(\boldsymbol{v}^h\right) = r_2^h \end{cases}. \tag{5.15}$$

In order to correct the approximated solution $\boldsymbol{v}^h$ on the fine grid, one needs to compute the error $\boldsymbol{e}^h$. However, the error cannot be computed directly. Since high frequency components of the error in pre-smoothing step have already been removed by the smoother, we can transfer

the following nonlinear system to the coarse grid as follows

$$\begin{cases} \underbrace{\mathcal{N}_1^h\left(\boldsymbol{v}^h+\boldsymbol{e}^h\right)}_{\mathcal{N}_1^h(\boldsymbol{u}^h)} = \underbrace{r_1^h+\mathcal{N}_1^h\left(\boldsymbol{v}^h\right)}_{g_1^h} \\ \underbrace{\mathcal{N}_2^h\left(\boldsymbol{v}^h+\boldsymbol{e}^h\right)}_{\mathcal{N}_2^h(\boldsymbol{u}^h)} = \underbrace{r_2^h+\mathcal{N}_2^h\left(\boldsymbol{v}^h\right)}_{g_2^h} \end{cases} \rightarrow \begin{cases} \underbrace{\mathcal{N}_1^H\left(\boldsymbol{v}^H+\boldsymbol{e}^H\right)}_{\mathcal{N}_1^H(\boldsymbol{u}^H)} = \underbrace{r_1^H+\mathcal{N}_1^H\left(\boldsymbol{v}^H\right)}_{g_1^H} \\ \underbrace{\mathcal{N}_2^H\left(\boldsymbol{v}^H+\boldsymbol{e}^H\right)}_{\mathcal{N}_2^H(\boldsymbol{u}^H)} = \underbrace{r_2^H+\mathcal{N}_2^H\left(\boldsymbol{v}^H\right)}_{g_2^H} \end{cases} \tag{5.16}$$

where $H = 2h$ is the new cell size $H_1 \times H_2$ with $H_1 \geq h_1$ and $H_2 \geq h_2$ and $g_l^H \neq 0$ on the coarse grid. After the nonlinear residual equation system on the coarse grid (5.16) have been solved with a method of our choice, the coarse-grid correction $\boldsymbol{e}^H = \boldsymbol{u}^H - \boldsymbol{v}^H$ is then interpolated back to the fine grid $\boldsymbol{e}^h$ that can now be used for updating the approximated solution $\boldsymbol{v}^h$ on the fine grid, i.e. $\boldsymbol{v}_{new}^h = \boldsymbol{v}^h + \boldsymbol{e}^h$ (**coarse-grid correction step**). The last step for a FAS multigrid is to perform the smoother again to remove high frequency parts of the interpolated error (**post-smoothing step**).

In our FAS multigrid for diffusion image registration, standard coarsening is used in computing the coarse-grid domain Ω_H by doubling the grid size in each space direction, i.e. $h \rightarrow 2h = H$. For intergrid transfer operators between Ω_h and Ω_H, the averaging and bilinear interpolation techniques are used for the restriction and interpolation operators denoted respectively by I_h^H and I_H^h; see more details in [12, 67, 134, 139, 140]. In order to compute the coarse-grid operator of $\mathcal{N}_l^h\left(\boldsymbol{u}^h\right)$ consisting of two parts: $\mathcal{L}^h\left(u_l^h\right)$ and $f_l^h\left(u_1^h, u_2^h\right)$, a so-called *discretisation coarse grid approximation* (DCA) is performed [12, 19, 134, 140]. The idea is to rediscretise the Euler-Lagrange directly. In the case of $\mathcal{L}^h\left(u_l^h\right)$, the corresponding coarse-grid part $\mathcal{L}^H\left(u_l^H\right)$ is obtained by the restriction of u_l^h and a simple adaptation of the grid size to the discretised Laplacian. For $f_l^h\left(u_1^h, u_2^h\right)$, we first use the restriction operator with both components of the deformation field $\boldsymbol{u}^h$, u_1^h and u_2^h, and the given images, R^h and T^h, and then compute the corresponding coarse-grid part $f_l^H\left(u_1^H, u_2^H\right)$. To solve (5.14) numerically, our FAS multigrid is applied recursively down to the coarsest grid consisting of a small number of grid points, typically 4×4, and may be summarised as follows:

Algorithm 5.2.1 (FAS Multigrid Algorithm)

Denote FAS multigrid parameters as follows:

ν_1 pre-smoothing steps on each level

ν_2 post-smoothing steps on each level

μ the number of multigrid cycles on each level (μ= 1 for V−cycling and μ= 2 for W−cycling).

(Here we present the V−cycle with μ= 1.)

α regularisation parameter

ω relaxation parameter

$GSiter$ the maximum number of iterations using a smoother

$$\left[v_1^h, v_2^h\right] \leftarrow FASCYC\left(v_1^h, v_2^h, \mathcal{N}_1^h, \mathcal{N}_2^h, g_1^h, g_2^h, R^h, T^h, \nu_1, \nu_2, \alpha, \omega, GSiter\right)$$

- If Ω_h = *coarset grid* ($|\Omega_h| = 4 \times 4$), solve (5.14) using time-marching techniques such as semi-implicit or AOS scheme (§5.2.3) and then stop. Else continue with following step.
- Pre-smoothing:
  For $z = 1$ to ν_1, $\left[v_1^h, v_2^h\right] \leftarrow Smoother\left(v_1^h, v_2^h, g_1^h, g_2^h, R^h, T^h, \alpha, \omega, GSiter\right)$
- Restriction to the coarse grid:
  $v_1^H \leftarrow I_h^H v_1^h, \quad v_2^H \leftarrow I_h^H v_2^h, \quad R^H \leftarrow I_h^H R^h, \quad T^H \leftarrow I_h^H T^h$
- Set the initial solution for the coarse-grid problem:
  $\left[\overline{v}_1^H, \overline{v}_2^H\right] \leftarrow \left[v_1^H, v_2^H\right]$
- Compute the new right-hand side for the coarse-grid problem:
  $g_1^H \leftarrow I_h^H\left(g_1^h - \mathcal{N}_1^h\left(v_1^h, v_2^h\right)\right) + \mathcal{N}_1^H\left(v_1^H, v_2^H\right),$
  $g_2^H \leftarrow I_h^H\left(g_2^h - \mathcal{N}_2^h\left(v_1^h, v_2^h\right)\right) + \mathcal{N}_2^H\left(v_1^H, v_2^H\right)$
- Implement the FAS multigrid on the coarse-grid problem:
  for $z = 1$ to μ, $\left[v_1^H, v_2^H\right] \leftarrow FASCYC\left(v_1^H, v_2^H, \mathcal{N}_1^H, \mathcal{N}_2^H, g_1^H, g_2^H, R^H, T^H, \nu_1, \nu_2, \alpha, \omega, GSiter\right)$
- Add the coarse-grid corrections:
  $v_1^h \leftarrow v_1^h + I_H^h\left(v_1^H - \overline{v}_1^H\right), \quad v_2^h \leftarrow v_2^h + I_H^h\left(v_2^H - \overline{v}_2^H\right)$
- Post-smoothing:
  For $z = 1$ to ν_2, $\left[v_1^h, v_2^h\right] \leftarrow Smoother\left(v_1^h, v_2^h, g_1^h, g_2^h, R^h, T^h, \alpha, \omega, GSiter\right)$

For practical applications our FAS-NMG method is stopped if the maximum number of V− or W−cycles ε_1 is reached (usually $\varepsilon_1 = 20$), the mean of the relative residuals obtained from the Euler-Lagrange equations (5.14) is smaller than a small number $\varepsilon_2 > 0$ (typically $\varepsilon_2 = 10^{-8}$ for a convergent test and only $\varepsilon_2 = 10^{-2}$ for a practical application), the relative reduction of the dissimilarity is smaller than some $\varepsilon_3 > 0$ (we usually assign $\varepsilon_3 = 0.20$ meaning that the relative reduction of the dissimilarity would decrease about 80%), or the change in two consecutive steps of the data/fitting term $\mathcal{D}$ is smaller than a small number $\varepsilon_4 > 0$ (typically $\varepsilon_4 = 10^{-6}$). A pseudo-code implementation of our FAS-NMG method is then summarised in the following algorithm:

Algorithm 5.2.2 (FAS Multigrid Method)

$$\boldsymbol{v}^h \leftarrow FASMG\left(\boldsymbol{v}^h, \alpha, \overrightarrow{\varepsilon}\right)$$

- Select α, $\overrightarrow{\varepsilon} = (\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4)$ and initial guess solutions $\boldsymbol{v}_{initial}^h = \left(v_1^h, v_2^h\right)^\top$ on the finest grid
- Set $K = 0$, $(\boldsymbol{v}^h)^K = \boldsymbol{v}_{initial}^h$, $\widetilde{\varepsilon}_2 = \varepsilon_2 + 1$, $\widetilde{\varepsilon}_3 = \varepsilon_3 + 1$, and $\widetilde{\varepsilon}_4 = \varepsilon_4 + 1$
- While ($K < \varepsilon_1$ AND $\widetilde{\varepsilon}_2 > \varepsilon_2$ AND $\widetilde{\varepsilon}_3 > \varepsilon_3$ AND $\widetilde{\varepsilon}_4 > \varepsilon_4$)
  - $(\boldsymbol{v}^h)^{K+1} = \left[v_1^h, v_2^h\right] \leftarrow FASCYC\left(v_1^h, v_2^h, \mathcal{N}_1^h, \mathcal{N}_2^h, g_1^h, g_2^h, R^h, T^h, \nu_1, \nu_2, \alpha, \omega, GSiter\right)$
  - $\widetilde{\varepsilon}_2 = mean\{||g_l^h - \mathcal{N}_l^h((\boldsymbol{v}^h)^{K+1})||_2 / ||g_l^h - \mathcal{N}_l^h\left(\boldsymbol{v}_{initial}^h\right)||_2 |\ l = 1, 2\}$
  - $\widetilde{\varepsilon}_3 = \mathcal{D}^h(R^h, T^h_{(\boldsymbol{v}^h)^{K+1}}) / \mathcal{D}^h(R^h, T^h)$,
    [Recall that $\mathcal{D}^h(R^h, T^h_{(\cdot)}) \sim \frac{h_1 h_2}{2} ||R^h, T^h_{(\cdot)}||_2^2$]
  - $\widetilde{\varepsilon}_4 = |\mathcal{D}^h(R^h, T^h_{(\boldsymbol{v}^h)^{K+1}}) - \mathcal{D}^h(R^h, T^h_{(\boldsymbol{v}^h)^K})|$
  - $K = K + 1$
- end

As is well known, in addition to restriction and interpolation operators, the above Algorithm 5.2.1 requires a suitable *smoother* based on some iterative relaxation method which is often the decisive factor in determining whether or not a multigrid algorithm converges. This issue will be discussed next after we review the linear multigrid method.

The linear multigrid method for (5.5)

For a nonlinear problem, a linearisation approach can lead to a coupled outer-inner iteration method with the inner solver provided by a linear multigrid method. For (5.5), the outer iteration is introduced either in a GN step [65, 83, 89] or in terms of the semi-implicit time marching scheme [73, 131] as follows

$$\begin{cases} (\mathbf{I} - \alpha\tau \sum_{l=1}^{2} \mathbf{A}_l) u_1^{(k+1)} = (u_1^{(k)} - \tau f_1(u_1^{(k)}, u_2^{(k)})) \\ (\mathbf{I} - \alpha\tau \sum_{l=1}^{2} \mathbf{A}_l) u_2^{(k+1)} = (u_2^{(k)} - \tau f_2(u_1^{(k)}, u_2^{(k)})) \end{cases}, \tag{5.17}$$

which is a system of two linear elliptic PDEs. Finally each outer step k is solved iteratively by a linear multigrid. In order to reduce the number of outer steps, a scale space framework described in [78] can be used for adapting automatically the registration parameters τ and α. Although, this numerical scheme is very accurate for providing visually pleasing registration results, we found experimentally that it is quite slow in fulfilling the necessary condition for being a minimiser of the variational problem represented by (5.1), i.e. in achieving convergence, because the linear system has to be solved many times with changing the right-hand side of (5.17); see Tables 5.2 – 5.3. This is a convenient way of using a multigrid method but is not as optimal as a nonlinear multigrid method.

The nonlinear multigrid method for (5.5)

The above introduced FAS-NMG algorithm can be readily applied to (5.5). The choice of a suitable smoother is a key for fast convergence. Below we briefly review four types of smoothers that have been or can be used for diffusion image registration:

1) The Newton-Gauss-Seidel relaxation smoother. This was used in Gao et al. [54] for (5.5). Although, there are no numerical results for the convergence of the FMG technique in their work, we found with several tests that this kind of smoothers provides visually pleasing registration results within a few multigrid steps. However, it does not perform well as a good smoother in leading to the convergence of the FAS-NMG technique. Note that this smoother can be derived directly from (5.13).

2) The fixed-point (FP) iteration based smoothers. The simple linearised iterations by the following

$$\begin{cases} -\alpha\Delta u_1^{[\nu+1]} = -\left(I'_{\boldsymbol{u}^{[\nu]}} - R\right) \partial_{u_1} I'_{\boldsymbol{u}^{[\nu]}} \\ -\alpha\Delta u_2^{[\nu+1]} = -\left(I'_{\boldsymbol{u}^{[\nu]}} - R\right) \partial_{u_2} I'_{\boldsymbol{u}^{[\nu]}} \end{cases} \tag{5.18}$$

as discussed in [104, p.79] have been used by researchers; we shall name this method the standard FP (SDFP) scheme. This SDFP approach encounters a singular system for all fixed-point step ν due to Neumann's boundary conditions. Without any special treatment for overcoming the singularity, we found that simple smoothers such as the Jacobi-, GS-, or successive overrelaxation (SOR)-type methods usually fail to lead to convergence of the FAS-NMG technique; however we discuss ways of improving this idea below.

3) The augmented system technique for the SDFP scheme. This was introduced by Frohn-Schauf et al. [51, 53]. The idea is to convert the singular system to a nonsingular one by augmenting extra equations. However, it may not lead to satisfaction of the necessary (first order) condition of a minimiser of the variational problem represented by (5.1). In details, this smoother builds on the above SDFP method (5.18) with an initial guess $\boldsymbol{u}^{[0]}$

$$\begin{cases} -\alpha\mathcal{L}(u_1^{[\nu+1]}) = g_1 - f_1(u_1^{[\nu]}, u_2^{[\nu]}) = G_1^{[\nu]} \\ -\alpha\mathcal{L}(u_2^{[\nu+1]}) = g_2 - f_2(u_1^{[\nu]}, u_2^{[\nu]}) = G_2^{[\nu]} \end{cases}. \qquad \text{(SDFP)} \tag{5.19}$$

Recall that $g_1 = g_2 = 0$ on the finest grid in multigrid setting and the symbol h and $(\cdot)_{i,j}$ in (5.8) are dropped for simplicity. As already mentioned above, the resulting system is singular and symmetric in case of Neumann's boundary conditions approximated by (5.9). The reason for singularity is row sum zero in boundary points, i.e. the constant functions lie in the kernel of $\mathcal{L}$. In this case the discrete system has a solution if and only if the discrete compatibility condition

$$\sum_{i,j=1}^{n_1,n_2} (G_l^{[\nu]})_{i,j} = 0 \qquad \text{for } l = 1, 2$$

is satisfied [12, 51, 53, 134]. Obviously, this condition fails when the given images are substantially different. Recognizing the above difficulties, Frohn-Schauf et al. [51, 53] solved a nearby problem created by a simple modification, which guarantees that discrete solutions exist for each fixed-point problem from replacing $G_l^{[\nu]}$ $(l = 1, 2)$ by

$$\widetilde{G}_l^{[\nu]} = G_l^{[\nu]} - \frac{\left\langle G_l^{[\nu]}, \mathbb{I} \right\rangle}{\langle \mathbb{I}, \mathbb{I} \rangle},$$

where $\mathbb{I}$ is the $n_1 \times n_2-$vector $(1, \ldots, 1)$. Note that if $\boldsymbol{u}^{[\nu+1]}$ solves the new discrete system

$$\begin{cases} -\alpha\mathcal{L}(u_1^{[\nu+1]}) = \widetilde{G}_1^{[\nu]} \\ -\alpha\mathcal{L}(u_2^{[\nu+1]}) = \widetilde{G}_2^{[\nu]} \end{cases}, \tag{5.20}$$

then $\boldsymbol{u}^{[\nu+1]} + \boldsymbol{c}$ also solves the same problem for any $\boldsymbol{c}$. This means that the solution is not unique. In order to determine the unique solution of the discrete system given by (5.20), they put a constraint on $\boldsymbol{u}^{[\nu+1]}$. This can be done by applying the zero-mean condition,

$$\sum_{i,j=1}^{n_1,n_2} (u_l^{[\nu+1]})_{i,j} = \left\langle u_l^{[\nu+1]}, \mathbb{I} \right\rangle = c_l = 0 \qquad \text{for } l = 1, 2. \tag{5.21}$$

We shall denote the above method by MSDFP-FS (a modified SDFP scheme due to Frohn-Schauf et al.).

4) The modified standard FP scheme. Following the same idea of overcoming singularities, below, we consider 2 alternative ways of modifying the SDFP method (5.18)

$$\begin{cases} -\alpha\mathcal{L}(u_1^{[\nu+1]}) + c_1 u_1^{[\nu+1]} = G_1^{[\nu]} + c_1 u_1^{[\nu]} \\ -\alpha\mathcal{L}(u_2^{[\nu+1]}) + c_2 u_2^{[\nu+1]} = G_2^{[\nu]} + c_2 u_2^{[\nu]} \end{cases}, \qquad \text{(MSDFP-1)} \tag{5.22}$$

$$\begin{cases} (-\alpha\mathcal{L} + \epsilon) u_1^{[\nu+1]} = G_1^{[\nu]} \\ (-\alpha\mathcal{L} + \epsilon) u_2^{[\nu+1]} = G_2^{[\nu]} \end{cases}, \qquad \text{(MSDFP-2)} \tag{5.23}$$

where c_l ($l = 1, 2$) and ϵ are positive real numbers (very small number for ϵ). We note first that the modified SDFP method of the first type (MSDFP-1) given by (5.22) can be viewed as a semi-implicit time marching scheme when $c_1 = c_2 = c > 0$ is interpreted to be the time-step, as used by Zikic et al. and Modersitzki [104, p.80]. Second, the second type (MSDFP-2) represented by (5.23) is the simplest way to stabilise the SDFP method by adding the small number ϵ to all diagonal elements of $\mathcal{L}$, as used by Brito-Loeza and Chen [13] in a different context. Finally, we note that determining the optimal values for the FP parameters c_1, c_2 (or c), and ϵ in automatic procedures leading to convergence of the FP method (and a multigrid technique) is not straightforward for real-world applications because tuning is needed for different registration problems.

In other tests, we found experimentally that although the approximation solutions obtained from the MSDFP-FS scheme are visually pleasing, they may not fulfill the necessary condition for being a minimiser of the variational problem (5.1); see Table 5.2. The reason is that this numerical scheme solves a nearby problem for each fixed-point step ν by changing the right hand side of (5.19) subject to the zero mean condition (5.21). These difficulties have motivated us to develop the new smoother in the next section.

5.3 The proposed algorithm with a new and robust smoother

To first design a better smoother for (5.5), we have to re-consider the nonlinear terms in (5.19) in a new FP scheme. Once this is done, a basic linear iterative solver such as the Jacobi, GS or SOR method for each corresponding system may be used. Then to improve the model robustness, we use a multi-resolution idea to choose the regularisation parameter α.

5.3.1 The new FP smoother

Our idea of a new FP scheme is different from the SDFP scheme (5.18) and its variants, by aiming for full implicitness in both regulariser and data terms. This yields

$$\begin{cases} -\alpha\mathcal{L}(u_1^{[\nu+1]}) + f_1(u_1^{[\nu+1]}, u_2^{[\nu+1]}) = g_1 \\ -\alpha\mathcal{L}(u_2^{[\nu+1]}) + f_2(u_1^{[\nu+1]}, u_2^{[\nu+1]}) = g_2 \end{cases}. \tag{5.24}$$

Next, we linearise the data term $f_l(u_1^{[\nu+1]}, u_2^{[\nu+1]})$ via a first-order Taylor's expansion of form (for $l = 1, 2$)

$$\begin{aligned} f_l(u_1^{[\nu+1]}, u_2^{[\nu+1]}) &\sim f_l(u_1^{[\nu]}, u_2^{[\nu]}) + \partial_{u_1} f_l(u_1^{[\nu]}, u_2^{[\nu]})\delta u_1^{[\nu]} + \partial_{u_2} f_l(u_1^{[\nu]}, u_2^{[\nu]})\delta u_2^{[\nu]}, \\ &= f_l(u_1^{[\nu]}, u_2^{[\nu]}) + \sigma_{l1}^{[\nu]}\delta u_1^{[\nu]} + \sigma_{l2}^{[\nu]}\delta u_2^{[\nu]}, \\ &= f_l(u_1^{[\nu]}, u_2^{[\nu]}) + \sigma_{l1}^{[\nu]}(u_1^{[\nu+1]} - u_1^{[\nu]}) + \sigma_{l2}^{[\nu]}(u_2^{[\nu+1]} - u_2^{[\nu]}) \end{aligned} \tag{5.25}$$

where

$$\sigma_{l1}(\boldsymbol{u}^{[\nu]}) = \partial_{u_1} f_l(u_1^{[\nu]}, u_2^{[\nu]}) = (\partial_{u_l} T_{\boldsymbol{u}^{[\nu]}})(\partial_{u_1} T_{\boldsymbol{u}^{[\nu]}}) + (T_{\boldsymbol{u}^{[\nu]}} - R)(\partial_{u_1 u_l} T_{\boldsymbol{u}^{[\nu]}}),$$

and

$$\sigma_{l2}(\boldsymbol{u}^{[\nu]}) = \partial_{u_2} f_l(u_1^{[\nu]}, u_2^{[\nu]}) = (\partial_{u_l} T_{\boldsymbol{u}^{[\nu]}})(\partial_{u_2} T_{\boldsymbol{u}^{[\nu]}}) + (T_{\boldsymbol{u}^{[\nu]}} - R)(\partial_{u_2 u_l} T_{\boldsymbol{u}^{[\nu]}}).$$

By (5.25), it leads to the semi-implicitly FP iteration step in terms of a 2×2 matrix as follows

$$\mathbf{N}[\boldsymbol{u}^{[\nu]}]\boldsymbol{u}^{[\nu+1]} = \mathbf{G}[\boldsymbol{u}^{[\nu]}], \qquad \text{(RFP)} \tag{5.26}$$

where RFP refers to the robust FP scheme (to be tested shortly), $\boldsymbol{u}^{[\nu+1]} = (u_1^{[\nu+1]}, u_2^{[\nu+1]})^\top$,

$$\mathbf{N}[\boldsymbol{u}^{[\nu]}] = \begin{bmatrix} -\alpha\mathcal{L} + \sigma_{11}(\boldsymbol{u}^{[\nu]}) & \sigma_{12}(\boldsymbol{u}^{[\nu]}) \\ \sigma_{21}(\boldsymbol{u}^{[\nu]}) & -\alpha\mathcal{L} + \sigma_{22}(\boldsymbol{u}^{[\nu]}) \end{bmatrix},$$
$$\mathbf{G}[\boldsymbol{u}^{[\nu]}] = \begin{pmatrix} G_1^{[\nu]} + \sigma_{11}(\boldsymbol{u}^{[\nu]})u_1^{[\nu]} + \sigma_{12}(\boldsymbol{u}^{[\nu]})u_2^{[\nu]} \\ G_2^{[\nu]} + \sigma_{21}(\boldsymbol{u}^{[\nu]})u_1^{[\nu]} + \sigma_{22}(\boldsymbol{u}^{[\nu]})u_2^{[\nu]} \end{pmatrix}.$$

To solve (5.26), we adopt the *block or pointwise collective Gauss-Seidel* (PCGS) relaxation method, i.e. all difference equations are updated simultaneously. A PCGS step is then given by

$$(\boldsymbol{u}^{[\nu+1]})_{i,j}^{[k+1]} = (\mathbf{N}[\boldsymbol{u}^{[\nu]}]_{i,j})^{-1}(\mathbf{G}[\boldsymbol{u}^{[\nu]}])_{i,j}^{[k+1/2]}, \tag{5.27}$$

where

$$\mathbf{N}[\boldsymbol{u}^{[\nu]}]_{i,j} = \begin{bmatrix} (\alpha/h_1^2)(\Sigma)_{i,j} + (\sigma_{11}(\boldsymbol{u}^{[\nu]}))_{i,j} & (\sigma_{12}(\boldsymbol{u}^{[\nu]}))_{i,j} \\ (\sigma_{21}(\boldsymbol{u}^{[\nu]}))_{i,j} & (\alpha/h_1^2)(\Sigma)_{i,j} + (\sigma_{22}(\boldsymbol{u}^{[\nu]}))_{i,j} \end{bmatrix},$$
$$(\mathbf{G}[\boldsymbol{u}^{[\nu]}])_{i,j}^{[k+1/2]} = \begin{pmatrix} (G_1^{[\nu]})_{i,j} + (\sigma_{11}(\boldsymbol{u}^{[\nu]}))_{i,j}(u_1^{[\nu]})_{i,j} + (\sigma_{12}(\boldsymbol{u}^{[\nu]}))_{i,j}(u_2^{[\nu]})_{i,j} \\ +(\alpha/h_1^2)(\overline{\Sigma})_{i,j}(u_1^{[\nu+1]})_{i,j}^{[k+1/2]} \\ \\ (G_2^{[\nu]})_{i,j} + (\sigma_{21}(\boldsymbol{u}^{[\nu]}))_{i,j}(u_1^{[\nu]})_{i,j} + (\sigma_{22}(\boldsymbol{u}^{[\nu]}))_{i,j}(u_2^{[\nu]})_{i,j} \\ +(\alpha/h_1^2)(\overline{\Sigma})_{i,j}(u_2^{[\nu+1]})_{i,j}^{[k+1/2]} \end{pmatrix}.$$

To gain more efficiency, one may introduce a relaxation parameter $\omega \in (0, 2)$ and iterate the ω−PCGS steps by

$$(\boldsymbol{u}^{[\nu+1]})_{i,j}^{[k+1]} = (1-\omega)\,(\boldsymbol{u}^{[\nu+1]})_{i,j}^{[k]} + \omega\underbrace{(\mathbf{N}[\boldsymbol{u}^{[\nu]}]_{i,j})^{-1}(\mathbf{G}[\boldsymbol{u}^{[\nu]}])_{i,j}^{[k+1/2]}}_{\text{original PCGS result}}. \tag{5.28}$$

We note first that the proposed smoother shows the interaction between the actual FP iteration (5.26) that overcomes the nonlinearity of the operator $\mathcal{N}_l$ at each outer step ν and the ω−PCGS method (5.28) that solves the resulting linear system of equations at each corresponding inner step k. Second, instead of solving the linear system of equations using the inner solver (5.28) with very high precision, it can perform only a *few iterations* to obtain an approximation solution at each outer step. Evidently, this procedure leads to a slight difference of convergence in the FP scheme when the proposed smoother is used as a stand-alone solver, whereas the computational costs significantly reduce; see Figure 5.1 (a). Moreover, the relaxation parameter ω also has a strong influence on the convergence speed. As the stand-alone solver of (5.14) we usually use $\omega > 1$, typically $\omega = 1.85$, because it results in speeding up the convergence compared with the PCGS approach ($\omega = 1$); see Figure 5.1 (b). For our multigrid algorithm, however, we use the local Fourier analysis and several experiments to select the optimal value of ω; see §5.3.2 later. Finally we remark that other iterative techniques such as the line relaxation techniques or the preconditioned conjugate gradient method may also be used as an inner solver. However, the ω−PCGS relaxation method appears a cheaper option for practical applications.

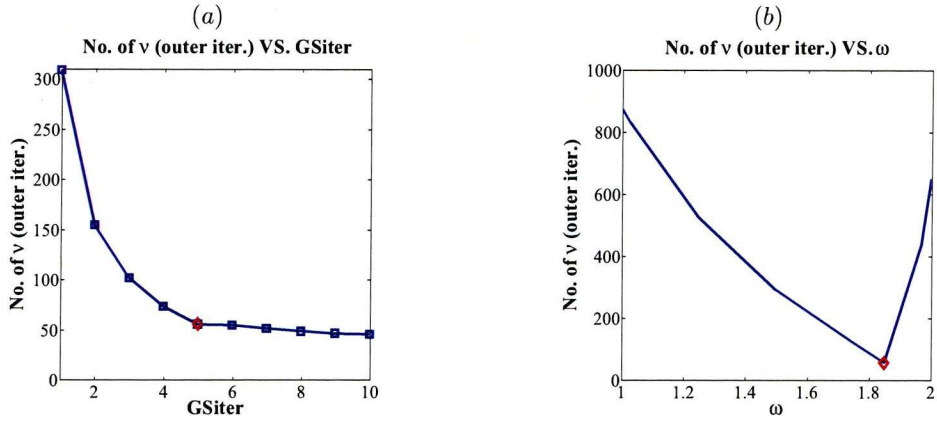


Figure 5.1: Number of outer iterations ν in (5.26) used to drop the mean of relative residuals of (5.24) to 10^{-8} for different values of (a) *GSiter* and (b) ω at a fixed value of $\alpha = 0.1$ for processing the registration problem in Examples 1 as shown in Figure 5.3 $(a)-(b)$ on a 32×32 grid. The red diamond indicates the optimal choice in each plot.

Implementation of our proposed smoother (5.26) based on the $\omega-$PCGS method (5.28) on a fine grid can be summarised as follows:

Algorithm 5.3.1 (Our Proposed Smoother)

Denote by

α regularisation parameter

ω relaxation parameter

GSiter the maximum number of $\omega-$PCGS iterations

$$[v_1^h, v_2^h] \leftarrow Smoother\left(v_1^h, v_2^h, g_1^h, g_2^h, R^h, T^h, \alpha, \omega, GSiter\right)$$

- Use input parameters to compute $(\sigma_{lm}[\boldsymbol{v}^h])_{i,j}$, $(\mathbf{G}[\boldsymbol{v}^h])_{i,j}$, and $(\mathbf{N}[\boldsymbol{v}^h]_{i,j})^{-1}$ for $l, m = 1, 2$, $1 \le i \le n_1$, and $1 \le j \le n_2$ (Here $(\boldsymbol{v}^h)_{i,j} = ((v_1^h)_{i,j}, (v_2^h)_{i,j})^\top$)
- Perform $\omega-$PCGS steps
  - for $k = 1 : GSiter$
    - for $i = 1 : n_1$
      - for $j = 1 : n_2$
        - Compute $(\boldsymbol{v}^h)_{i,j}^{[k+1]} = ((v_1^h)_{i,j}^{[k+1]}, (v_2^h)_{i,j}^{[k+1]})^\top)$ using (5.28)
      - end
    - end
  - end

We remark that the first-order Taylor's expansion of

$$\begin{aligned} T_{\boldsymbol{u}^{[\nu+1]}} &= T_{\boldsymbol{u}^{[\nu+1]}}(\mathbf{x}) = T(\mathbf{x} + \boldsymbol{u}^{[\nu+1]}) = T(\mathbf{x} + \boldsymbol{u}^{[\nu]} + \delta\boldsymbol{u}^{[\nu]}) \\ &\approx T(\mathbf{x} + \boldsymbol{u}^{[\nu]}) + \partial_{u_1} T(\mathbf{x} + \boldsymbol{u}^{[\nu]})\delta u_1^{[\nu]} + \partial_{u_2} T(\mathbf{x} + \boldsymbol{u}^{[\nu]})\delta u_2^{[\nu]}, \end{aligned} \tag{5.29}$$

is used to derive the FP schemes in a different context for a different technique; see [17, 18, 43]. We also wish to remark that the above quantities $\sigma_{l1}(\boldsymbol{u}^{[\nu]})$, $\sigma_{l2}(\boldsymbol{u}^{[\nu]})$ may be refined to derive a cheaper implementation than our FP method (5.26). We note first that $\sigma_{21}(\boldsymbol{u}^{[\nu]}) = \sigma_{12}(\boldsymbol{u}^{[\nu]})$. Second in order to have a simple and stable numerical scheme as pointed out by several works; see e.g. [89] and [104, p. 56/79], we approximate $\sigma_{lm}(\boldsymbol{u}^{[\nu]})$ by $\sigma_{lm}(\boldsymbol{u}^{[\nu]}) = (\partial_{u_l} I'_{\boldsymbol{u}^{[\nu]}})(\partial_{u_m} I'_{\boldsymbol{u}^{[\nu]}})$ for $m = 1, 2$ since the image difference $I'_{\boldsymbol{u}^{[\nu]}} - R$ becomes small for well registered images and the second-order derivatives of $I'_{\boldsymbol{u}^{[\nu]}}$ $(\partial_{u_m u_l} I'_{\boldsymbol{u}^{[\nu]}})$, a problematic part of $\sigma_{lm}(\boldsymbol{u}^{[\nu]})$, are very sensitive to noise and are hard to estimate robustly. Finally, we note that if discrete image gradient $\partial_{u_l} I'_{\boldsymbol{u}^{[\nu]}}$ does not vanish at one point, the system matrix of these linearised equations is strictly or irreducibly diagonally dominant. This guarantees the existence of a unique solution of each linearised system and global convergence of the Jacobi and GS iterations [117, 121].

Below we analyse the smoothing property of our proposed FP smoother (5.26) based on the ω−PCGS method (5.28).

5.3.2 Local Fourier analysis (LFA)

LFA is a powerful tool to analyse the smoothing properties of iterative algorithms used in MG methods. Although LFA was originally developed for discrete linear operators with constant coefficients on infinite grids, it can also be applied to more general nonlinear equations with varying coefficients such as the discrete version of (5.5). To this end, first an infinite grid is assumed to eliminate the effect of boundary conditions and second it is also assumed that the discrete nonlinear operator can be linearised (by freezing coefficients) and replaced locally by a new operator with constant coefficients [134]. This approach has proved to be very useful in the understanding of MG methods when solving nonlinear problems; see for instance [6, 7, 13, 22, 23, 65, 61, 69, 89, 126] for interesting examples and discussions.

Measure of h-ellipticity

It is well known that h-ellipticity is crucial for multigrid methods to be effective. It is often used to decide whether or not pointwise error smoothing procedures (e.g. our proposed smoother (5.26) based on (5.28)) can be constructed for the discrete operator under consideration. To this end, we shall show that the linearised system $\mathbf{N}_h[\overline{\boldsymbol{u}}^h]\boldsymbol{u}^h = \mathbf{G}_h[\overline{\boldsymbol{u}}^h]$ in (5.26) at some outer step provides a sufficient amount of h-ellipticity in a similar way as shown in [65, 89, 134, 140] for a discrete system of PDEs. Here $\boldsymbol{u}^h$ and $\overline{\boldsymbol{u}}^h$ denote the exact solution and the current approximation and $\mathbf{N}_h[\overline{\boldsymbol{u}}^h]$ and $\mathbf{G}_h[\overline{\boldsymbol{u}}^h]$ the resulting discrete operators from the linearisation at $\overline{\boldsymbol{u}}^h$. For simplicity, our analysis is carried out over the infinite grid

$$\Omega_h^\infty = \{\mathbf{x} \in \Omega | \mathbf{x} = (x_{1_i}, x_{2_j})^\top = ((2i-1)\,h/2, (2j-1)\,h/2)^\top,\ i, j \in \mathbb{Z}^2\} \tag{5.30}$$

where $h = 1/n$ denotes the mesh parameter.

Let $\boldsymbol{\varphi}_h(\boldsymbol{\theta}, \mathbf{x}) = \exp(\mathbf{i}\boldsymbol{\theta}\mathbf{x}/h) \cdot \widehat{\mathbf{I}}$ be grid functions, where $\widehat{\mathbf{I}} = (1,1)^\top$, $\boldsymbol{\theta} = (\theta_1, \theta_2)^\top \in \boldsymbol{\Theta} = (-\pi, \pi]^2$, $\mathbf{x} \in \Omega_h^\infty$, and $\mathbf{i} = \sqrt{-1}$. It is important to remark that due to the locality nature

of LFA, our analysis applies to each grid point separately, i.e. we consider the local discrete system $\mathbf{N}_h(\xi)\boldsymbol{u}^h = \mathbf{G}_h(\xi)$ centered and defined only within a small neighborhood of each grid point $\xi = (i, j)$ and $\boldsymbol{u}^h(\xi) = [u_1^h(\xi), u_2^h(\xi)]$. Applying the discrete operator $\mathbf{N}_h(\xi)$ to the grid functions $\boldsymbol{\varphi}_h(\boldsymbol{\theta}, \mathbf{x})$, i.e. $\mathbf{N}_h(\xi)\boldsymbol{\varphi}_h(\boldsymbol{\theta}, \mathbf{x}) = \widehat{\mathbf{N}}_h(\xi, \boldsymbol{\theta})\boldsymbol{\varphi}_h(\boldsymbol{\theta}, \mathbf{x})$, yields the Fourier symbol as follows:

$$\widehat{\mathbf{N}}_h(\xi, \boldsymbol{\theta}) = \begin{bmatrix} -\alpha\widehat{\mathcal{L}}^h(\boldsymbol{\theta}) + \sigma_{11}(\xi) & \sigma_{12}(\xi) \\ \sigma_{21}(\xi) & -\alpha\widehat{\mathcal{L}}^h(\boldsymbol{\theta}) + \sigma_{22}(\xi) \end{bmatrix} \tag{5.31}$$

(see details related to Fourier symbols of systems of PDEs in [134, 140]). Here $\widehat{\mathcal{L}}^h(\boldsymbol{\theta})$ denotes the Fourier symbol of the discrete Laplacian operator $\widehat{\mathcal{L}}^h$. Following [134, 140], the measure of h-ellipticity is defined via $\widehat{\mathbf{N}}_h(\xi, \boldsymbol{\theta})$ as follows:

$$E_h(\mathbf{N}_h(\xi)) = \frac{\min\{|\det(\widehat{\mathbf{N}}_h(\xi, \boldsymbol{\theta}))| : \boldsymbol{\theta} \in \boldsymbol{\Theta}_{\text{high}}\}}{\max\{|\det(\widehat{\mathbf{N}}_h(\xi, \boldsymbol{\theta}))| : \boldsymbol{\theta} \in \boldsymbol{\Theta}\}}, \tag{5.32}$$

where $\boldsymbol{\Theta}_{\text{high}} = \boldsymbol{\Theta} \backslash (-\pi/2, \pi/2]^2$ denotes the range of high frequencies and

$$\det(\widehat{\mathbf{N}}_h(\xi, \boldsymbol{\theta})) = \alpha^2(\widehat{\mathcal{L}}^h(\boldsymbol{\theta}))^2 + \alpha c_1(\widehat{\mathcal{L}}^h(\boldsymbol{\theta})) + c_2 \tag{5.33}$$

represents the determinant of $\widehat{\mathbf{N}}_h(\xi, \theta)$ where

$$c_1 = -(\sigma_{11}(\xi) + \sigma_{22}(\xi)) \text{ and } c_2 = \sigma_{11}(\xi)\sigma_{22}(\xi) - \sigma_{12}(\xi)\sigma_{21}(\xi).$$

According to the well-known results, we obtain

$$-\widehat{\mathcal{L}}^h(\boldsymbol{\theta}) = (2/h^2)\left(2 - (\cos\theta_1 + \cos\theta_2)\right), \min_{\boldsymbol{\theta} \in \boldsymbol{\Theta}_{\text{high}}}(-\widehat{\mathcal{L}}^h(\boldsymbol{\theta})) = -\widehat{\mathcal{L}}^h(-\pi/2, 0) = 2/h^2$$

and $\max_{\theta \in \boldsymbol{\Theta}}(-\widehat{\mathcal{L}}^h(\boldsymbol{\theta})) = -\widehat{\mathcal{L}}^h(\pi, \pi) = 8/h^2$. Therefore,

$$E_h(\mathbf{N}_h(\xi)) = \frac{2\alpha + c_1h^2 + (c_2h^4/2\alpha)}{32\alpha + 4c_1h^2 + (c_2h^4/2\alpha)} \tag{5.34}$$

and

$$\lim_{h \to 0} E_h(\mathbf{N}_h(\xi)) = \frac{1}{16} \tag{5.35}$$

bounded away from 0 for all possible choices $\alpha, h > 0$ and for all possible values of $\sigma_{11}(\xi)$, $\sigma_{12}(\xi)$, $\sigma_{21}(\xi)$, and $\sigma_{22}(\xi)$ (i.e. the results do not depend on the given images) over the whole discrete domain Ω_h.

As a result, it can be expected that the discrete system $\mathbf{N}_h[\overline{\boldsymbol{u}}^h]\boldsymbol{u}^h = \mathbf{G}_h[\overline{\boldsymbol{u}}^h]$ is appropriate to pointwise error smoothing procedures like our proposed smoother (5.26) combining with the ω−PCGS method (5.28).

5.3.3 Smoothing analysis for the proposed smoother

A robust and potential smoother has to take care of the high-frequency components of the error between the exact solution and the current approximation since the low-frequency components becomes the high-frequency components on coarser grids and they cannot be reduced on coarser

grids by the coarse-gird correction procedure. A quantitative measure of the smoothing efficiency for a given algorithm is the *smoothing factor* denoted by μ_{loc} from a LFA and numerically computed for test problems, which is defined as the worst asymptotic error reduction, by performing one smoother step, of all high-frequency error components [134, 140]. Below we shall analyse the smoothing properties of the proposed smoother via (5.26) and (5.28).

As pointed out in many cases of nonlinear operators with varying coefficients by [6, 7, 13, 22, 23, 65, 61, 69, 89, 126], the smoothing factor is **x**-dependent. Therefore, it is customary to look for the maximum over the local smoothing factors of the frozen operator $\mathbf{N}_h(\xi)$, i.e.

$$\mu^*_{\text{loc}} = \max_{\varsigma \in \Omega_h} \mu_{\text{loc}} \tag{5.36}$$

To determine μ_{loc} we consider again the local discrete system $\mathbf{N}_h(\xi)\boldsymbol{u}^h(\xi) = \mathbf{G}_h(\xi)$. By using the splitting $\mathbf{N}_h(\xi) = \mathbf{N}_h^{[+]}(\xi) + \mathbf{N}_h^{[0]}(\xi) + \mathbf{N}_h^{[-]}(\xi)$, it is possible to write the local inner iterations of (5.26) (for $\omega = 1$) as

$$\mathbf{N}_h^{[+]}(\xi)\overline{\boldsymbol{u}}^h_{new}(\xi) + \mathbf{N}_h^{[0]}(\xi)\overline{\boldsymbol{u}}^h_{new}(\xi) + \mathbf{N}_h^{[-]}(\xi)\overline{\boldsymbol{u}}^h_{old}(\xi) = \mathbf{G}_h(\xi) \tag{5.37}$$

where $\overline{\boldsymbol{u}}^h_{old}(\xi)$ and $\overline{\boldsymbol{u}}^h_{new}(\xi)$ stand for the approximations to $\boldsymbol{u}^h(\xi)$ before and after the inner smoothing step, respectively and

$$\mathbf{N}_h^{[+/0/-]}(\xi) = \begin{bmatrix} (\mathbf{N}_h^{[+/0/-]}(\xi))_{1,1} & (\mathbf{N}_h^{[+/0/-]}(\xi))_{1,2} \\ (\mathbf{N}_h^{[+/0/-]}(\xi))_{2,1} & (\mathbf{N}_h^{[+/0/-]}(\xi))_{2,2} \end{bmatrix}$$

For a specification of this splitting, we use the stencil notation as follows:

$$\mathcal{L}^h_{[+]} = \frac{\alpha}{h^2}\begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix}, \quad \mathcal{L}^h_{[0]} = \frac{\alpha}{h^2}\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathcal{L}^h_{[-]} = \frac{\alpha}{h^2}\begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix},$$

$$(\mathbf{N}_h^{[+/-]}(\xi))_{l,m} = \begin{cases} \mathcal{L}^h_{[+/-]} & \text{for } l = m \\ 0, & \text{for } l \neq m \end{cases}, \quad (\mathbf{N}_h^{[0]}(\xi))_{l,m} = \kappa_{l,m}\mathcal{L}^h_0 \quad (l,m = 1,2),$$

and

$$\varsigma = \begin{bmatrix} \varsigma_{1,1} & \varsigma_{1,2} \\ \varsigma_{2,1} & \varsigma_{2,2} \end{bmatrix} = \begin{bmatrix} \Sigma + (h^2/\alpha)\sigma_{11}(\xi) & (h^2/\alpha)\sigma_{12}(\xi) \\ (h^2/\alpha)\sigma_{21}(\xi) & \Sigma + (h^2/\alpha)\sigma_{22}(\xi) \end{bmatrix}.$$

By subtracting (5.37) from $\mathbf{N}_h(\xi)\boldsymbol{u}^h(\xi) = \mathbf{G}_h(\xi)$ and defining $\overline{\boldsymbol{e}}^h_{new}(\xi) = \boldsymbol{u}^h(\xi) - \boldsymbol{u}^h_{new}(\xi)$ and $\overline{\boldsymbol{e}}^h_{old}(\xi) = \boldsymbol{u}^h(\xi) - \boldsymbol{u}^h_{old}(\xi)$ we obtain the local system of error equations

$$\mathbf{N}_h^{[+]}(\xi)\overline{\boldsymbol{e}}^h_{new}(\xi) + \mathbf{N}_h^{[0]}(\xi)\overline{\boldsymbol{e}}^h_{new}(\xi) + \mathbf{N}_h^{[-]}(\xi)\overline{\boldsymbol{e}}^h_{old}(\xi) = 0$$

or

$$\overline{\boldsymbol{e}}^h_{new}(\xi) = -\left[\mathbf{N}_h^{[0]}(\xi) + \mathbf{N}_h^{[+]}(\xi)\right]^{-1}[\mathbf{N}_h^{[-]}(\xi)]\overline{\boldsymbol{e}}^h_{old}(\xi) = \mathbf{S}_h(\xi)\overline{\boldsymbol{e}}^h_{old}(\xi) \tag{5.38}$$

where $\mathbf{S}_h(\xi)$ is the amplification factor. The effect of $\mathbf{S}_h(\xi)$ on the grid functions $\varphi_h(\boldsymbol{\theta}, \mathbf{x})$ within $\boldsymbol{\Theta}_{\text{high}} = \boldsymbol{\Theta}\backslash[-\pi/2, \pi/2]^2$ will determine the smoothing properties of the PCGS method (5.27).

For the ω−PCGS approach (5.28), the amplification factor denoted by $\mathbf{S}_h(\xi, \omega)$ can be defined in a similar way as (5.38) and its Fourier symbol is given by

$$\widehat{\mathbf{S}}_h(\xi, \boldsymbol{\theta}, \omega) = [\widehat{\mathbf{N}}_h^0(\xi, \boldsymbol{\theta}) + \omega\widehat{\mathbf{N}}_h^+(\xi, \boldsymbol{\theta})]^{-1}[(1-\omega)\widehat{\mathbf{N}}_h^0(\xi, \boldsymbol{\theta}) - \omega\widehat{\mathbf{N}}_h^-(\xi, , \boldsymbol{\theta})] \in \mathbb{C}^{2\times 2} \tag{5.39}$$

Here the Fourier symbols of $\mathbf{N}_h^{[+/0/-]}(\xi)$ are

$$\widehat{\mathbf{N}}_h^{[+/0/-]}(\xi,\boldsymbol{\theta}) = \begin{bmatrix} (\widehat{\mathbf{N}}_h^{[+/0/-]}(\xi,\boldsymbol{\theta}))_{1,1} & (\widehat{\mathbf{N}}_h^{[+/0/-]}(\xi,\boldsymbol{\theta}))_{1,2} \\ (\widehat{\mathbf{N}}_h^{[+/0/-]}(\xi,\boldsymbol{\theta}))_{2,1} & (\widehat{\mathbf{N}}_h^{[+/0/-]}(\xi,\boldsymbol{\theta}))_{2,2} \end{bmatrix}. \tag{5.40}$$

Therefore, the local smoothing factor is

$$\mu_{\text{loc}} = \sup\{|\rho(\widehat{\mathbf{S}}_h(\xi,\boldsymbol{\theta},\omega))| : \boldsymbol{\theta} \in \boldsymbol{\Theta}_{\text{high}}\} \tag{5.41}$$

where ρ indicates the spectral radius of $\widehat{\mathbf{S}}_h(\xi,\boldsymbol{\theta},\omega)$. Recall that

$$(\mathbf{N}_h^{[+]}(\xi,\boldsymbol{\theta}))_{l,m} = \begin{cases} -\frac{\alpha}{h^2}(\exp(-\mathbf{i}\theta_1) + \exp(-\mathbf{i}\theta_2))) & \text{for } l = m \\ 0, & \text{for } l \neq m \end{cases}, \tag{5.42}$$

$$(\mathbf{N}_h^{[-]}(\xi,\boldsymbol{\theta}))_{l,m} = \begin{cases} -\frac{\alpha}{h^2}(\exp(\mathbf{i}\theta_1) + \exp(\mathbf{i}\theta_2))) & \text{for } l = m \\ 0, & \text{for } l \neq m \end{cases} \tag{5.43}$$

$$(\mathbf{N}_h^{[0]}(\xi,\boldsymbol{\theta}))_{l,m} = \frac{\alpha}{h^2}\varsigma_{l,m} \tag{5.44}$$

will be used to compute (5.39).

To select the optimal value of ω and test our smoother we consider one set of medical images as shown respectively in Figure 5.3 $(a)-(b)$ on a 32×32 grid. Figure 5.2 shows the smoothing factors of the proposed smoother (5.26) based on the $\omega-$PCGS approach (5.28) at different values of ω. It indicates that the optimal value ω providing $\mu_{\text{loc}}^* \approx 0.5$ is not exactly 1 but very close to 1, typically $\omega = 0.9725$.

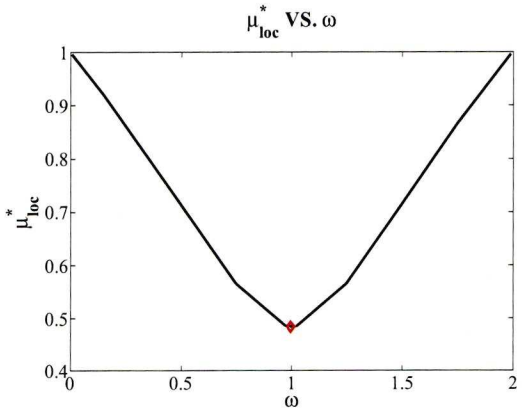


Figure 5.2: Smoothing factors μ_{loc}^* at a fixed value of $\alpha = 0.1$ after 5 outer iterations with $GSiter = 5$ by the proposed smoother (5.26) based on the $\omega-$PCGS approach (5.28) with different values of ω for the registration problem in Examples 1 as shown in Figure 5.3 $(a)-(b)$ on a 32×32 grid. The red diamond indicates the optimal value of ω.

We remark that we have to deal with a type of anisotropy in solving the linearised system $\mathbf{N}_h[\overline{\boldsymbol{u}}^h]\boldsymbol{u}^h = \mathbf{G}_h[\overline{\boldsymbol{u}}^h]$ (5.26). This anisotropy is not global but local, the *jumping coefficients* introduced by $\sigma_{l,m}(\overline{\boldsymbol{u}}^h)$. We have $\sigma_{l,m}(\overline{\boldsymbol{u}}^h) \neq 0$ at places that correspond to regions where the transformed image $T_{\overline{\boldsymbol{u}}^h}^h$ changes locally, e.g. at edges. However, from practical experience this leads to moderate jumps in the coefficients and then the smoothing factors shown in Figure 5.2 are not rigorously justified. They can be considered as a heuristic but reliable estimate

for actual smoothing properties since we only have moderate jumps. We conducted several numerical experiments to back up our results from smoothing analysis in §5.5.1.

A nonlinear multigrid algorithm with automatic choice of α

As is typical of Tikhonov regularisation, the energy functional $\mathcal{J}_\alpha$ in (5.1) has a regularisation parameter α. To provide well matched images, we have to carefully select α because it is in general unknown a priori. In order to find a suitable α automatically, we follow the 'cooling' ('continuation') process suggested in [29, 32, 66, 65, 78, 92, 108]. The basic idea is to start with a high initial value of α and then slowly reduce α such that the obtained solution can be used to be an excellent starting point for the next minimisation problem in order to decrease $\mathcal{J}_\alpha$. An alternative approach can be the L-curve method.

Consider the discrete version of the minimisation problem (5.1) with the same notation

$$\min_{\boldsymbol{u}}\{\mathcal{J}_\alpha(\boldsymbol{u}) = \mathcal{D}(R, T(\boldsymbol{u})) + \alpha\mathcal{R}(\boldsymbol{u})\}. \tag{5.45}$$

Here $\mathcal{D} = \mathcal{D}^{\text{SSD}}$ and $\mathcal{R} = \mathcal{R}^{\text{diff}}$. Let α_1 be the initial value, which is sufficiently large. At the $(s+1)$th step we set $\alpha^{(s+1)} = \eta\alpha^{(s)} \in [\alpha_0, \alpha_1]$, where $\eta \in (0,1)$ is a constant, usually chosen to be about 0.5, and α_0 is a small positive number, e.g. 5×10^{-5}. Subsequently, we apply $\alpha^{(s+1)}$ and the initial guess solution obtained by the previous iteration $\boldsymbol{u}_{initial}^{(s+1)} = \boldsymbol{u}^{(s)}$ with the associated inner loop to obtain the minimum $\boldsymbol{u}^{(s+1)}$ within some tolerance. As mentioned in [66], since the functional $\mathcal{J}_\alpha$ is changing at each outer loop iteration, the demand of decreasing the value of the same functional is not reasonable. Then, the solution $\boldsymbol{u}^{(s+1)}$ and parameter $\alpha^{(s+1)}$ are acceptable if they satisfy the so-called *consistent condition*:

$$\mathcal{J}_{\alpha^{(s+1)}}(\boldsymbol{u}^{(s+1)}) = \mathcal{D}(\boldsymbol{u}^{(s+1)}) + \alpha^{(s+1)}\mathcal{R}(\boldsymbol{u}^{(s+1)}) < \mathcal{J}_{\alpha^{(s+1)}}(\boldsymbol{u}^{(s)}) = \mathcal{D}(\boldsymbol{u}^{(s)}) + \alpha^{(s+1)}\mathcal{R}(\boldsymbol{u}^{(s)}).$$

However, if this condition is not satisfied, we increase η (usually to 0.9) and re-start the step. Our experience suggests that the stopping criterion given by

$$\frac{\left\|\boldsymbol{u}^{(s+1)} - \boldsymbol{u}^{(s)}\right\|_{l_2}}{\max\{\left\|\boldsymbol{u}^{(s+1)}\right\|_{l_2}, \left\|\boldsymbol{u}^{(s)}\right\|_{l_2}\}} < \delta \tag{5.46}$$

is suitable, where $\delta > 0$ is small (normally set to 10^{-3}).

Finally, we summarise this process as follows:

Algorithm 5.3.2 (Multigrid Image Registration Through Cooling)

$$[\boldsymbol{v}^*, \alpha^*] \leftarrow \textit{cooling}\left(\boldsymbol{v}^{(0)}, \alpha^{(0)}, \overrightarrow{\varepsilon}\right)$$

- Set $s = 1$, $\boldsymbol{v}^{(s)} = \boldsymbol{v}^{(0)}$, $\alpha^{(s)} = \alpha^{(0)}$, $\eta = 0.5$.
- Outer iteration: For $s = 1, 2, 3, ...$
  - 1. Set $\alpha^{(s+1)} = \eta\alpha^{(s)}$ in $[5 \times 10^{-5}, \alpha^{(s)}]$
  - 2. Inner iteration: $\boldsymbol{v}_{new} \leftarrow FASMG\left(\boldsymbol{v}^{(s)}, \alpha^{(s+1)}, \overrightarrow{\varepsilon}\right)$
  - 3. If $\mathcal{J}_{\alpha^{(s+1)}}(\boldsymbol{v}_{new}] < \mathcal{J}_{\alpha^{(s+1)}}(\boldsymbol{v}^{(s)})$
    - 3.1 Set $\boldsymbol{v}^{(s+1)} = \boldsymbol{v}_{new}$, $\eta = 0.5$, $s = s + 1$, and go to 4

    Else
    - 3.2 Set $\eta = 0.9$, and go to 4
  - 4. Check for convergence using the criterion (5.46)

    If not satisfied, then return to 1, else, exit to the next step to stop.
- Set $\boldsymbol{v}^* = \boldsymbol{v}_{new}$ and $\alpha^* = \alpha^{(s)}$.

In order to save computational work for high-resolution digital images, the low-tolerance $\overrightarrow{\varepsilon}_{lo} = (2, 10^{-4}, 0.1, 10^{-4})$ is applied to reduce the accumulated costs in each minimisation problem. Then our first algorithm, namely *a robust diffusion image registration* (RDR) *approach*, can be stated as follows:

Algorithm 5.3.3 (The basic RDR method)

1. Input $\overrightarrow{\varepsilon}_{lo}$. Set $\alpha = 1$ (optional). Set $\overrightarrow{\varepsilon}_{hi} = (20, 10^{-8}, 0.10, 10^{-8})$ (high-tolerance)
2. Obtain the optimal regularisation parameter α (through cooling) via Algorithm 5.3.2:
   - $[\boldsymbol{v}^{(0)}, \alpha] \leftarrow cooling\,(\boldsymbol{v}, \alpha, \overrightarrow{\varepsilon}_{lo})$.
3. Solve the discrete minimisation problem (5.45) on the finest level using the found α:
   - $\boldsymbol{v} \leftarrow FASMG(\boldsymbol{v}^{(0)}, \alpha, \overrightarrow{\varepsilon}_{hi})$

Although the above algorithm enables us to find a good α, the cost of re-solving the same problem repeatedly is expensive. We propose to use a hierarchy of L grids (with level L the finest and level 1 the coarsest one) using a multi-resolution idea to gain efficiency while finding an effective α. Firstly we shall seek the optimal α on the coarsest level 1 with the grid size of 32×32 only (believed to be coarse enough) and secondly we use the multilevel continuation idea [65] to provide the initial guesses for the next finer level.

Algorithm 5.3.4 (Multilevel grid continuation for optimal $\alpha$and reliable initial solution)

$[\boldsymbol{v}^{[lev]}, \alpha^{[lev]}] \leftarrow RDR\_multiresolution(\boldsymbol{v}^{[lev]}, \alpha^{[lev]}, lev, \overrightarrow{\varepsilon})$

- If $lev = 1$
  - $\boldsymbol{v}^{[lev]} = 0$
  - $\alpha^{[lev]} = C$ [$C > 0$ should be large enough e.g. $C = 100$]
  - $[\boldsymbol{v}^{[lev]}, \alpha^{[lev]}] \leftarrow cooling(\boldsymbol{v}^{[lev]}, \alpha^{[lev]}, \overrightarrow{\varepsilon})$
- Else
  - $\boldsymbol{v}^{[lev-1]} = (I_h^H v_1^{[lev]}, I_h^H v_2^{[lev]})^\top$
  - $\left[\boldsymbol{v}^{[lev-1]}, \alpha^{[lev-1]}\right] \leftarrow RDR\_multiresolution(\boldsymbol{v}^{[lev-1]}, \alpha^{[lev-1]}, lev - 1, \overrightarrow{\varepsilon})$
  - $\boldsymbol{v}^{[lev]} = (I_H^h v_1^{[lev-1]}, I_H^h v_2^{[lev-1]})^\top$
  - $\alpha^{[lev]} = 4\alpha^{[lev-1]}$ [Recall that $\alpha^{[lev]} = \overline{\alpha} n_{lev}^2$ and $n_{lel} = 2n_{lel-1}$]
  - $\boldsymbol{v}^{[lev]} \leftarrow FASMG(\boldsymbol{v}^{[lev]}, \alpha^{[lev]}, \overrightarrow{\varepsilon})$
- Endif

Finally the overall procedure of finding an optimal α and then starting a nonlinear multigrid method to solve (5.1) is summarised below as Algorithm 5.3.5:

Algorithm 5.3.5 (The refined RDR multi-resolution method)

---

1. Input $\overrightarrow{\varepsilon}_{lo}$ and $\overrightarrow{\varepsilon}_{hi}$.
2. Obtain the optimal regularisation parameter α (through cooling) and a good initial solution (through multi-resolution) $\boldsymbol{v}^{(0)}$ via Algorithm 5.3.4:
   - $[\boldsymbol{v}^{(0)}, \alpha] \leftarrow RDR\_multiresolution(\boldsymbol{v}^{[L]}, \alpha^{[L]}, L, \overrightarrow{\varepsilon}_{lo})$
3. Solve the minimisation problem (5.45) on the finest level $lev = L$ using the found α and the initial guess solution $\boldsymbol{v}^{(0)}$:
   - $\boldsymbol{v}^{[lev]} \leftarrow FASMG(\boldsymbol{v}^{(0)}, \alpha, \overrightarrow{\varepsilon}_{hi})$

5.4 An application of (5.25) with the curvature model

To test the robustness of our numerical algorithm for other registration models, in this section, we shall examine our second test model, namely, the curvature image registration model, as introduced by Fischer and Modersitzki [47]; see also [48, 49, 89, 91, 104].

The curvature model. Based on an approximation of the mean curvature of the surface of u_l, Fischer–Modersitzki's curvature approach aims to find a reasonable deformation field $\boldsymbol{u}$ that minimises the following functional [47, 48]

$$\min_{\boldsymbol{u}}\{\mathcal{J}_\alpha(\boldsymbol{u}) = \mathcal{D}^{\mathrm{SSD}}(\boldsymbol{u}) + \alpha\mathcal{R}^{\mathrm{FMcurv}}(\boldsymbol{u})\},$$

where

$$\mathcal{R}^{\mathrm{FMcurv}}(\boldsymbol{u}) = \frac{1}{2}\sum_{l=1}^{2}\int_\Omega (\widehat{\kappa}_M(u_l))^2 d\mathbf{x} = \frac{1}{2}\sum_{l=1}^{2}\int_\Omega (\Delta u_l)^2 dx. \tag{5.47}$$

This leads to the Euler-Lagrange system of two fourth-order nonlinear PDEs:

$$\begin{cases} f_1(\boldsymbol{u}) + \alpha\Delta^2 u_1 = 0 \\ f_2(\boldsymbol{u}) + \alpha\Delta^2 u_2 = 0 \end{cases} \quad (\textit{Fischer–Modersitzki's} \text{ curvature model}) \tag{5.48}$$

subject to the special boundary conditions $\nabla u_l = 0$, $\nabla\Delta u_l = 0$ on $\partial\Omega$, for $l = 1, 2$. We remark that the use of second-order derivatives in the energy functional (5.47) not only provides smoother deformation fields $\boldsymbol{u}$ than those of (5.3), but also allows for an automatic rigid alignment. Here u_l is understood as a surface in $\mathbb{R}^3$ represented by $(x_1, x_2, u_l(x_1, x_2))$, where initially $u_l(x_1, x_2) = 0$, with the *mean curvature* of the surface of u_l is given by

$$\kappa_M(u_l) = \nabla\cdot\frac{\nabla u_l}{\sqrt{1+|\nabla u_l|^2}} = \frac{(1+u_{l_{x_1}}^2)u_{l_{x_1x_1}} - 2u_{l_{x_1}}u_{l_{x_2}}u_{l_{x_1x_2}} + (1+u_{l_{x_2}}^2)u_{l_{x_2x_2}}}{(1+u_{l_{x_1}}^2+u_{l_{x_2}}^2)^{3/2}}. \tag{5.49}$$

Observe that $|\nabla u_l| \approx 0$ yields $\kappa_M(u_l) \approx \widehat{\kappa}_M(u_l) = \Delta u_l$.

Due to the biharmonic operator which appears in (5.48) it is known that standard iterative methods lead to poor multigrid efficiency. Therefore, it is a common way to split up the biharmonic operator into a system of two Poisson-type equations; see [12, 67, 73, 89, 134, 139, 140]. Based on this splitting idea, (5.48) can be converted to the following system using

additional unknown functions $v_1 = -\Delta u_1$ and $v_2 = -\Delta u_2$:

$$\begin{cases} -\Delta u_1 - v_1 = 0 \\ -\Delta u_2 - v_2 = 0 \\ f_1(\boldsymbol{u}) - \alpha \Delta v_1 = 0 \\ f_2(\boldsymbol{u}) - \alpha \Delta v_2 = 0 \end{cases} \tag{5.50}$$

subject to the boundary conditions transferred into $\nabla u_l = 0$ and $\nabla v_l = 0$ on $\partial\Omega$ where the data term $f_l(\boldsymbol{u})$ is given by (5.5).

To solve the above continuous system numerically, (5.50) is first discretised by the cell-centered finite difference scheme over the discrete domain

$$\Omega_h = \{\mathbf{x} \in \Omega | \mathbf{x} = (x_{1_i}, x_{2_j})^\top = ((2i-1)h/2, (2j-1)h/2),\ 1 \le i, j \le n\}$$

consist of $N = n^2$ cells of size $h \times h$ with grid spacing $h = (1/n, 1/n)$. Let $(z_{\widehat{l}}^h)_{i,j} = z_{\widehat{l}}^h(x_{1_i}, x_{2_j})$ denote the grid function for $\widehat{l} = 1, \ldots, 4$, where $z_{\widehat{l}}^h = u_l^h$ or v_l^h for $l = 1, 2$. Then, the discrete system of (5.50) at a grid point (i, j) is given by

$$\begin{cases} \mathcal{N}_1^h(\boldsymbol{z}^h)_{i,j} = -\mathcal{L}^h(u_1^h)_{i,j} - (v_1^h)_{i,j} = (g_1^h)_{i,j} \\ \mathcal{N}_2^h(\boldsymbol{z}^h)_{i,j} = -\mathcal{L}^h(u_2^h)_{i,j} - (v_2^h)_{i,j} = (g_2^h)_{i,j} \\ \mathcal{N}_3^h(\boldsymbol{z}^h)_{i,j} = f_1^h(u_1^h, u_2^h)_{i,j} - \alpha \mathcal{L}^h(v_1^h)_{i,j} = (g_3^h)_{i,j} \\ \mathcal{N}_4^h(\boldsymbol{z}^h)_{i,j} = f_2^h(u_1^h, u_2^h)_{i,j} - \alpha \mathcal{L}^h(v_2^h)_{i,j} = (g_4^h)_{i,j} \end{cases}, \tag{5.51}$$

where $(\boldsymbol{z}^h)_{i,j} = ((z_1^h)_{i,j}, (z_2^h)_{i,j}, (z_3^h)_{i,j}, (z_4^h)_{i,j})^\top = ((u_1^h)_{i,j}, (u_2^h)_{i,j}, (v_1^h)_{i,j}, (v_2^h)_{i,j})^\top$ and $(g_{\widehat{l}}^h)_{i,j} = 0$ $(\widehat{l} = 1, \ldots, 4)$ on the finest grid in multigrid setting. Recall that the discrete versions of $\mathcal{L}^h$ and $f_l^h(u_1^h, u_2^h)_{i,j}$ are given in the same way as represented by §5.2.2 and the approximations used in (5.51) need to be modified at grid points near the image boundary $\partial\Omega_h$ using the homogeneous Neumann boundary conditions approximated by one-side differences for boundary derivatives:

$$(z_{\widehat{l}}^h)_{i,1} = (z_{\widehat{l}}^h)_{i,2},\ (z_{\widehat{l}}^h)_{i,n} = (z_{\widehat{l}}^h)_{i,n-1},\ (z_{\widehat{l}}^h)_{1,j} = (z_{\widehat{l}}^h)_{2,j},\ (z_{\widehat{l}}^h)_{n,j} = (z_{\widehat{l}}^h)_{n-1,j}. \tag{5.52}$$

A robust smoother based on the linearisation scheme (5.25). As mentioned above, our aim is to apply our linearisation idea (5.25) in solving the equivalent system (5.50). This leads to the linearised system

$$\mathbf{N}^{\mathrm{Cv}}[\boldsymbol{z}^{[\nu]}]\boldsymbol{z}^{[\nu+1]} = \mathbf{G}^{\mathrm{Cv}}[\boldsymbol{z}^{[\nu]}], \tag{5.53}$$

where the symbols h and $(\cdot)_{i,j}$ in (5.51) are dropped for simplicity. Here

$$\mathbf{N}^{\mathrm{Cv}}[\boldsymbol{z}^{[\nu]}] = \begin{bmatrix} -\mathcal{L} & 0 & -1 & 0 \\ 0 & -\mathcal{L} & 0 & -1 \\ \sigma_{11}(\boldsymbol{u}^{[\nu]}) & \sigma_{12}(\boldsymbol{u}^{[\nu]}) & -\alpha\mathcal{L} & 0 \\ \sigma_{21}(\boldsymbol{u}^{[\nu]}) & \sigma_{22}(\boldsymbol{u}^{[\nu]}) & 0 & -\alpha\mathcal{L} \end{bmatrix} \tag{5.54}$$

and

$$\mathbf{G}^{\mathrm{Cv}}[\boldsymbol{z}^{[\nu]}] = \begin{pmatrix} g_1 \\ g_2 \\ g_3 - f_1(u_1^{[\nu]}, u_2^{[\nu]}) + \sigma_{11}(\boldsymbol{u}^{[\nu]})u_1^{[\nu]} + \sigma_{12}(\boldsymbol{u}^{[\nu]})u_2^{[\nu]} \\ g_4 - f_2(u_1^{[\nu]}, u_2^{[\nu]}) + \sigma_{21}(\boldsymbol{u}^{[\nu]})u_1^{[\nu]} + \sigma_{22}(\boldsymbol{u}^{[\nu]})u_2^{[\nu]} \end{pmatrix}. \tag{5.55}$$

As mentioned in §5.3.1, $\sigma_{21}^{[\nu]} = \sigma_{12}^{[\nu]}$ and $\sigma_{lm}^{[\nu]} = (\partial_{u_l} T(\boldsymbol{u}^{[\nu]}))(\partial_{u_m} T(\boldsymbol{u}^{[\nu]}))$ for $m = 1, 2$ are used to stabilise our numerical scheme. In order to solve (5.53), we again apply the $\omega-$PCGS as the inner solver and its kth step is updated by

$$(\boldsymbol{z}^{[\nu+1]})_{i,j}^{[k+1]} = (1-\omega)\,(\boldsymbol{z}^{[\nu+1]})_{i,j}^{[k]} + \omega(\mathbf{N}^{\mathrm{Cv}}[\boldsymbol{z}^{[\nu]}]_{i,j})^{-1}(\mathbf{G}^{\mathrm{Cv}}[\boldsymbol{z}^{[\nu]}])_{i,j}^{[k+1/2]}. \tag{5.56}$$

where

$$\mathbf{N}^{\mathrm{Cv}}[\boldsymbol{z}^{[\nu]}]_{i,j} = \begin{bmatrix} (\Sigma)_{i,j}/h^2 & 0 & -1 & 0 \\ 0 & (\Sigma)_{i,j}/h^2 & 0 & -1 \\ (\sigma_{11}(\boldsymbol{u}^{[\nu]}))_{i,j} & (\sigma_{12}(\boldsymbol{u}^{[\nu]}))_{i,j} & \alpha(\Sigma)_{i,j}/h^2 & 0 \\ (\sigma_{21}(\boldsymbol{u}^{[\nu]}))_{i,j} & (\sigma_{22}(\boldsymbol{u}^{[\nu]}))_{i,j} & 0 & \alpha(\Sigma)_{i,j}/h^2 \end{bmatrix} \tag{5.57}$$

$$(\Sigma)_{i,j} = \begin{cases} 2 \text{ at } (1,1),\ (n,1),\ (n,1), \text{ and } (n,n) \text{ (four corners)} \\ 3 \text{ for all } (1,j),\ (n,j),\ (i,1), \text{ and } (i,n) \text{ where } 2 \le i,j \le n-1 \text{ (boundary lines)} \\ 4 \text{ for all } (i,j) \text{ where } 2 \le i,j \le n-2 \text{ (interior points)} \end{cases}, \tag{5.58}$$

$$(\mathbf{G}^{\mathrm{Cv}}[\boldsymbol{z}^{[\nu]}])_{i,j}^{[k+1/2]} = \begin{pmatrix} (g_1)_{i,j} + (\overline{\Sigma})_{i,j}(u_1^{[\nu+1]})_{i,j}^{[k+1/2]} \\ \\ (g_2)_{i,j} + (\overline{\Sigma})_{i,j}(u_2^{[\nu+1]})_{i,j}^{[k+1/2]} \\ \\ (g_3)_{i,j} - f_1(u_1^{[\nu]}, u_2^{[\nu]})_{i,j} + (\sigma_{11}(\boldsymbol{u}^{[\nu]}))_{i,j}(u_1^{[\nu]})_{i,j} \\ +(\sigma_{12}(\boldsymbol{u}^{[\nu]}))_{i,j}(u_2^{[\nu]})_{i,j} + (\alpha/h^2)(\overline{\Sigma})_{i,j}(v_1^{[\nu+1]})_{i,j}^{[k+1/2]} \\ \\ (g_4)_{i,j} - f_2(u_1^{[\nu]}, u_2^{[\nu]})_{i,j} + (\sigma_{22}(\boldsymbol{u}^{[\nu]}))_{i,j}(u_2^{[\nu]})_{i,j} \\ +(\sigma_{21}(\boldsymbol{u}^{[\nu]}))_{i,j}(u_1^{[\nu]})_{i,j} + (\alpha/h^2)(\overline{\Sigma})_{i,j}(v_2^{[\nu+1]})_{i,j}^{[k+1/2]} \end{pmatrix}, \tag{5.59}$$

and

$$(\overline{\Sigma})_{i,j}(z_{\widehat{l}}^{[\nu+1]})_{i,j}^{[k+1/2]} = (z_{\widehat{l}}^{[\nu+1]})_{i+1,j}^{[k]} + (z_{\widehat{l}}^{[\nu+1]})_{i-1,j}^{[k+1]} + (z_{\widehat{l}}^{[\nu+1]})_{i,j+1}^{[k]} + (z_{\widehat{l}}^{[\nu+1]})_{i,j-1}^{[k+1]}. \tag{5.60}$$

With the above smoother, the curvature model can be solved using Algorithms 5.2.2 and 5.3.5.

Similarly to §5.3.2 one can show by the LFA that (i) $\lim_{h\to 0} E_h(\mathbf{N}_h^{\mathrm{Cv}}(\xi)) = \frac{1}{256}$, i.e. the linearised system (5.53) is $h-$elliptic; and (ii) $\omega \approx 1$ (typically $\omega = 0.9725$) provides the good smoothing properties ($\mu_{\mathrm{loc}}^* \approx 0.5$), where the effects of ω on smoothing factors of (5.53) in processing the registration problem shown in Figure 5.3 $(a)-(b)$ are almost identical with Figure 5.2. We also conducted several numerical tests to confirm that (5.53) is a robust smoother for the FAS-NMG method to solve the curvature model (5.48); see §5.5.2.

5.5 Numerical experiments

The main aim of this section is to show that our multigrid algorithms with the smoothers using (5.25) are effective and robust in leading to convergent multigrid methods in the FAS-NMG framework for both the diffusion and curvature models. In all experiments, the bilinear interpolation was used to compute the transformed template image $T(\boldsymbol{u})$.

5.5.1 The diffusion model

We first focus on the performance of Algorithm 5.3.5 for two sets of medical data, Examples[1] 1 and 2 shown respectively in Figure 5.3 $(a) - (b)$ and $(d) - (e)$. To this end, we consider its convergence behavior with different resolutions, and then show comparisons with several other methods. In all experiments for this section $\nu_1 = 5$, $\nu_2 = 5$, $GSiter = 5$, and $\omega = 0.9725$ were used in this test.

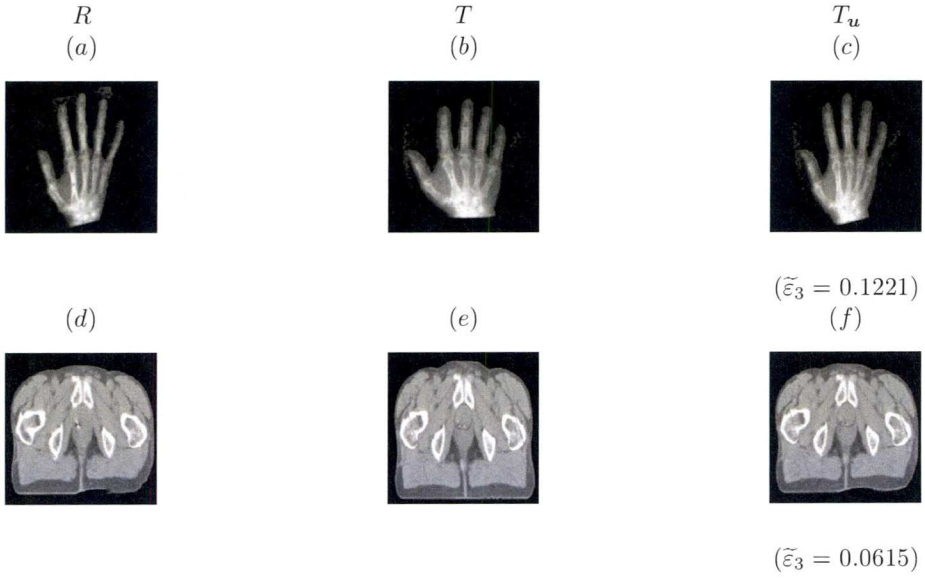


Figure 5.3: Registration results for X-ray and MRI images using the RDR method with Algorithms 5.2.2, 5.3.3, and 5.3.5. Left column: reference R, center column: template T, right column: the deformed template image $T(\boldsymbol{u})$ obtained from Algorithm 5.3.5.

$h-$independent convergence tests

One of the key properties of multigrid techniques is that their convergence does not depend on the number of grid points. Thus, in the first test we designed our experiments to investigate this property with Algorithms 5.2.2, 5.3.3, and 5.3.5, and to back up our theoretical results by LFA in §5.3.2. The number of multigrid steps (V-cycles) used to drop the mean of the relative residual below $\varepsilon_2 = 10^{-8}$, the relative reduction of dissimilarity, and run times (in seconds) are given in Table 5.1 with different sizes of grid points. The results show that all registration algorithms not only converge within a few multigrid steps, but they are also accurate because the dissimilarities between the reference and registered images have been reduced more than 87% for Example 1 and 94% for Example 2. For overall performance the experimental results suggest that Algorithm 5.3.5 would be preferred for practical applications because the multi-resolution idea used in the cooling process for α has been prove to be is very useful for initialisation.

[1]Source: http://www.math.mu-luebeck.de/safir/

	Algorithm 5.2.2 M/R/D/C	**Algorithm 5.3.3** M/R/D/IC/C	**Algorithm 5.3.5** M/R/D/IC/C
Example 1 :	$\alpha = 0.1000$		
$h = 1/128$	$10/2.1 \times 10^{-9}/0.1082/4.3$	$6/1.8 \times 10^{-9}/0.1082/7.9/10.1$	$7/3.4 \times 10^{-9}/0.1082/2.1/6.2$
$h = 1/256$	$11/3.6 \times 10^{-9}/0.1161/22.5$	$6/3.1 \times 10^{-9}/0.1161/41.2/54.2$	$7/5.1 \times 10^{-9}/0.1161/3.9/24.2$
$h = 1/512$	$11/2.8 \times 10^{-9}/0.1221/106.4$	$6/2.5 \times 10^{-9}/0.1221/180.1/231.9$	$6/7.5 \times 10^{-9}/0.1221/7.3/65.1$
$h = 1/1024$	$11/8.6 \times 10^{-9}/0.1239/472.6$	$6/3.8 \times 10^{-9}/0.1239/798.1/1049.3$	$6/8.2 \times 10^{-9}/0.1239/26.1/256.4$
Example 2 :	$\alpha = 0.1176$		
$h = 1/128$	$10/8.3 \times 10^{-9}/0.0522/4.2$	$5/1.5 \times 10^{-9}/0.0522/7.2/10.2$	$6/4.2 \times 10^{-9}/0.0552/2.9/4.9$
$h = 1/256$	$11/5.6 \times 10^{-9}/0.0582/22.9$	$6/2.6 \times 10^{-9}/0.0582/31.3/41.7$	$7/6.9 \times 10^{-9}/0.0582/3.0/17.9$
$h = 1/512$	$11/7.4 \times 10^{-9}/0.0615/108.3$	$7/5.4 \times 10^{-9}/0.0615/185.7/234.9$	$7/3.3 \times 10^{-9}/0.0615/12.5/79.1$
$h = 1/1024$	$11/3.1 \times 10^{-9}/0.0633/478.1$	$7/6.9 \times 10^{-9}/0.0633/550.1/943.1$	$7/5.0 \times 10^{-9}/0.0633/26.9/321.3$

Table 5.1: Registration results of Algorithms 5.2.2, 5.3.3, and 5.3.5 for Example 1 and 2 shown in Figure 5.3 $(a)-(b)$ and $(d)-(e)$. The letters 'M', 'R', 'D', 'C', and 'IC' mean the number of multigrid steps, the relative reduction of residual, the relative reduction of dissimilarity, the total run times, and the initial run times for determining the optimal α and initial guess $\boldsymbol{u}^{(0)}$, respectively.

Comparison with other multigrid methods

Methods by [46, 65, 89, 131, 145] are some existing unilevel or multigrid techniques used to solve the diffusion model.

In this section, we took Example 1 as shown in 5.3 $(a)-(b)$ to illustrate a comparison among our FAS-NMG method with Algorithm 5.2.2 and other six multigrid methods by starting with the fixed parameters $h = 1/256$, $\alpha = 1/8$ and $\boldsymbol{u}^{(0)} = 0$. Here we used $\tau = 10^{-2}$ and applied the so-called *multi-resolution technique* with the gradient descent methods of [46, 131] for fairness. The FMG or LMG methods in [65, 89, 131] were performed using two pre-smoothing and two-post smoothing steps with the pointwise GS smoothers until the mean of relative residuals below a user-supplied threshold ($tol_{\text{LMG}} = 10^{-4}$).

Table 5.2 summarises the results for all multigrid methods. As expected from the experiments, all methods are very fast and accurate in registering the given images because the dissimilarities between the reference and registered images have been reduced more than 80% within the first 20 iterations. However our method is not only the fastest way in solving the problem, but also in dropping the mean of the relative residuals below 10^{-8}.

Methods	M/R/D
Multi-resolution + AOS [46]	$20/7.1 \times 10^{-4}/0.1417$
Multi-resolution + FMG-V(2,2) [131]	$20/5.9 \times 10^{-4}/0.1428$
Multi-resolution + Gauss-Newton + LMG-V(2,2) [65, 89]	$20/6.3 \times 10^{-7}/0.1344$
FAS-NMG-V(5,5) + MSDFP-FS smoother adapted from [51, 53] (§5.2.4)	$20/2.6 \times 10^{-5}/0.1377$
FAS-NMG-V(5,5) + MSDFP-1 smoother adapted from [145, 104] (§5.2.4)	$18/4.5 \times 10^{-9}/0.1351$
FAS-NMG-V(5,5) + MSDFP-2 smoother adapted from [13] (§5.2.4)	$18/7.6 \times 10^{-9}/0.1383$
FAS-NMG-V(5,5) + RFP smoother (5.26) (§5.3.1)	$11/2.0 \times 10^{-9}/0.1329$

Table 5.2: A comparison among different multigrid methods by [46, 65, 89, 131, 145] to solve the diffusion model in the first 20 iterations. The letters 'M', 'R', and 'D' mean the number of iterations in dropping the mean of the relative residuals resulting form (5.1) to 10^{-8}, the mean of the relative residuals, and the relative reduction of dissimilarity, respectively. Our proposed multigrid method in the last row is the fastest way.

Comparison with the AOS method [46]

The AOS method is one of the most widely used gradient descent techniques for diffusion image registration in the unilevel framework [46, 90, 131]. We also take Example 1 as shown in 5.3 $(a) - (b)$ to illustrate a comparison with our FAS-NMG method. Table 5.3 summarises the results for Algorithm 5.2.2 and AOS methods with different numbers of grid points. We used Algorithm 5.2.2 with $\alpha = 1/8$ for fairness (as Algorithm 5.3.5 will be even better). That is, we started both methods with the same α and the same initial guess, $\boldsymbol{u}^{(0)} = 0$. For the AOS method, the time-step τ is required to be sufficiently small for each size of the problem. We used $\tau = 10^{-2}$ for $h = 1/128 - 1/1024$. As expected from the experiments, both methods are very accurate in registering the given images because the dissimilarities between the reference and registered images have been reduced more than 85%. However the AOS fails to drop the relative residual to 10^{-8} in a few time steps (even for large values of $\tau > 10^{-2}$) and the run times used by Algorithm 5.2.2 are much faster than those of the AOS technique in delivering the same level of the relative dissimilarity.

	Algorithm 5.2.2 M/R/D/C	**AOS** M/R/D/C
$h = 1/128$	$10/8.6 \times 10^{-9}/0.1248/4.2$ (0.07 mins)	$10000/*/0.1248/934.1$ ($>$ 15 mins)
$h = 1/256$	$11/2.0 \times 10^{-9}/0.1329/23.1$ (0.38 mins)	$10000/*/0.1329/5500.8$ ($>$ 1.5 hours)
$h = 1/512$	$11/9.4 \times 10^{-9}/0.1383/105.2$ (1.75 mins)	$10000/*/0.1383/2498.3$ ($>$ 6.9 hours)
$h = 1/1024$	$11/4.7 \times 10^{-9}/0.1404/427.6$ (7.96 mins)	$*/*/*/*$ ($>$ 12 hours)

Table 5.3: Registration results of Algorithm 5.2.2 and AOS method [46] for Example 1 shown in Figure 5.3 $(a) - (b)$. $*$ indicates either computation stopped after about 12 hours or failure in dropping the relative residual to 10^{-8} in 10000 iterations.

Comparison of multigrid methods with different smoothers

Our aim in this section is to show that we have proposed a robust smoother for the FAS-NMG technique in solving the discrete system represented in (5.14). To end this, we have conducted several experiments of our FAS multigrid with different kinds of smoothers:

(i) the proposed smoother based on the RFP method (5.26) represented by Algorithm 5.3.1 in §5.3

(ii) the GS smoother based on the SDFP method given by (5.19) in §5.2.4 (the standard FP method defined by [104, p. 79] with the standard linear (inner) solver)

(iii) the GS smoother based on the MSDFP-FS method given by (5.20) and (5.21) in §5.2.4 (a modified SDFP method adapted from the numerical techniques of Frohn-Schauf et al. [51, 53] with the standard linear solver)

(iv) the GS smoother based on MSDFP-1 method given by (5.22) in §5.2.4 (a modified SDFP method of the first type with the standard linear solver)

(v) the GS smoother based on MSDFP-2 method given by (5.23) in §5.2.4 (a modified SDFP method of the second type with the standard linear solver)

(vi) line relaxation smoothers based on the RFP method in §5.3 (the RFP method given by (5.26) with another kind of iterative linear solvers)

(vii) the Newton-Gauss-Seidel smoother used by Gao et al. [54] and given explicitly by Lu et al. [94] in (5.13) (an alternative way of choice of smoothers used in solving the discrete Euler-Lagrange equations represented by (5.14))

Omitting the computational results, we remark that these observations can be made:

a) The smoother in (ii) requires more multigrid cycles than the proposed smoother and may not lead to the convergence of the FAS-NMG technique for small values of α.

b) As expected, the results based on the smoother (iii) do not find a solution that is the necessary condition of the original variational problem (5.1), although the underlying MG performs better than with other smoothers (slightly less well than with our RFP smoother).

c) The smoother in (iv)-(v) may take many multigrid cycles, not leading to the convergence of the FAS-NMG technique when the fixed-point parameters c_1, c_2 (or c), and ϵ are not well-selected (i.e. the NMG becomes sensitive to these parameters).

d) As expected, line relaxation smoothers require less multigrid cycles but more computational costs than the proposed smoother.

e) The Newton-Gauss-Seidel smoother provides well matched images within a few multigrid steps, but it may require more multigrid cycles than the proposed smoother in leading to the convergence of the FAS-NMG technique.

5.5.2 The curvature model

In this section, we aim to show that the FAS-NMG method with the smoother (5.53) based on our linearisation idea (5.25) is effective and robust to solve the curvature model (5.48) within the multigrid framework similar to Algorithm 5.2.2.

To this end, we took only one data set of medical images, Examples 1, as shown in Figure 5.3 $(a)-(b)$. We first investigate the convergence behaviour and the registration accuracy of our FAS-NMG method with different sizes of image resolutions. Second we compare three numerical solution methods for solving the curvature model (5.48). In all experiments, $\Omega = [0,1]^2$, $V = [0,1]$, $\nu_1 = 10$, $\nu_2 = 10$, $PCGSiter = 10$, and $\omega = 0.9725$ were used.

$h-$independent convergence tests

In this test, we started the registration processes with $\alpha = 10^{-4}$, $\boldsymbol{u}^{(0)} = 0$, and $h = 1/128, ..., h = 1/1024$. The registered image is shown in Figure 5.4 (c). As expected from our LFA results,

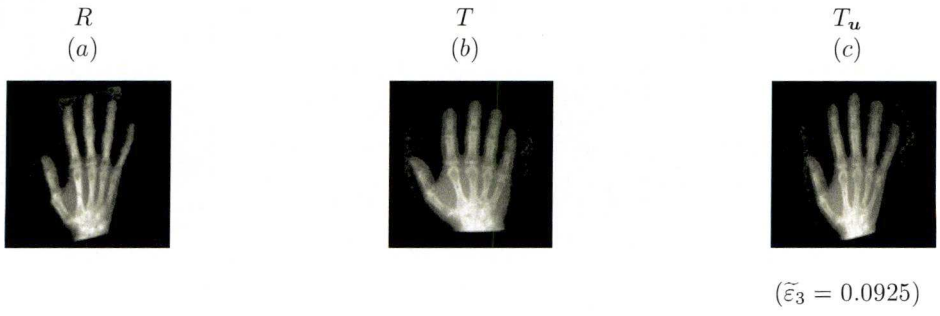


Figure 5.4: Results from Example 1 as shown in $(a) - (b)$ by the curvature model (5.48) using the FAS-NMG method with the smoother (5.53). Left to right: the reference R , the template T, and the registered image $T(\boldsymbol{u})$ by the curvature model (5.48).

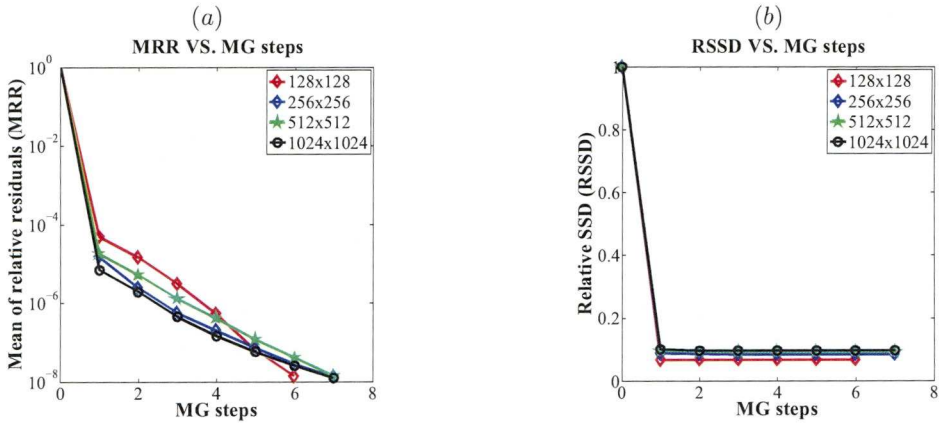


Figure 5.5: Results from Example 1 as shown by Figure 5.3 $(a) - (b)$ by the curvature model (5.48) using the FAS-NMG method with the smoother (5.53). Left to right: the histories of the mean of relative residuals (MRR) with respect to the MG steps and the histories of the relative SSD (RSSD) with respect to the MG steps.

Figure 5.5 $(a) - (b)$ shows that our FAS-NMG approach is $h-$independent. It takes only a few MG steps (almost the same number) to drop the mean of the relative residuals below 10^{-8}. Moreover, as shown in Figure 5.5 (b) only one MG step can reduce the dissimilarities between the reference and the registered image more than 90% for the given registration problem.

Comparison with the other two methods

In this section we aim to investigate the performance of the FAS-NMG method with the smoother (5.53) and those of other two methods: the FP method (5.53) (the smoother used as a stand-alone solver) and the DCT-based method by [48]. We note that the FT-based method by [91, 135] is faster than the DCT-based method with the ratio of 2.3 for directly solving each linear time-dependent problem resulting from (5.48). Thus, for this class of gradient descent

techniques it is enough to use only the DCT-based method, which is one of the most widely used techniques for the curvature model.

To this end, we started all methods with $h = 1/256$, $\alpha = 10^{-4}$ and $\boldsymbol{u}^{(0)} = 0$. For the DCT-based method, the time-step τ was selected to be $\tau = 10^{-2}$ (since the fixed parameters $1/h^4 = N^4 = 256^4$, $V = [0, 1]$ and $\alpha = 10^{-4}$ were used in the discrete system of (5.48), $\tau = 10^{-2}$ was a reasonable time step).

Compared with the results by other two methods, Figure 5.6 (b) shows that our FAS-NMG method is very fast in reducing the dissimilarities between the reference and registered images. Moreover, as shown in Figure 5.6 (a) it takes only a few steps to drop the mean of the relative residuals (MRR) below 10^{-8}. This is a remarkable result to conclude that its performance in solving the curvature model (5.48) is much more efficient than those of the other two methods.

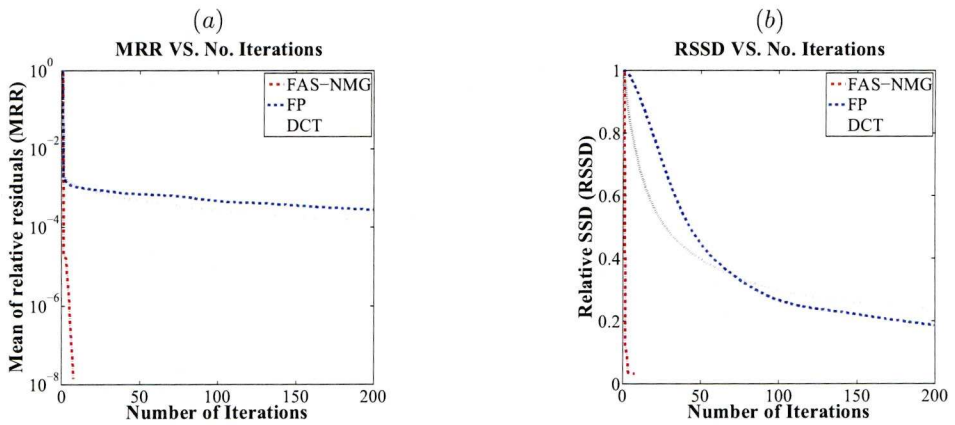


Figure 5.6: Results from Example 1 as shown by Figure 5.3 (a) – (b) by the curvature model (5.48) using three numerical solution methods: the FAS-NMG method with the smoother (5.53), the FP method (5.53), and the DCT-based method by [48]. Left to right: (a) the histories of the mean of relative residuals (MRR) with respect to the iteration steps and (b) the histories of the relative SSD (RSSD) with respect to the iteration steps.

From both tests in §5.5.1 and §5.5.2 they confirm that the smoothers based on our linearisation idea by (5.25) are effective and robust in leading to convergent multigrid methods for both the diffusion and curvature models.

5.6 Conclusions

In this chapter we first reviewed existing iterative methods for the diffusion model and then addressed the numerical problems of designing an optimal and efficient FAS-NMG technique. For the commonly used SSD model, we introduced a unified approach for designing FP type smoothers and used the LFA to analyse their smoothing properties using Fischer–Modersitzki's diffusion and curvature image registration models [46, 47]. In order to determine the optimal α, we applied the coarse-to-fine idea from the previous chapter with the proposed multigrid

approach and it appears to work well for a range of registration problems. Numerical experiments not only showed that the proposed multigrid approach is h-independent convergence, but it is also more effective than those in a large class of existing iterative methods developed by [46, 47, 48, 65, 89, 90, 131, 135, 145].

Chapter 6

A Discontinuity-Preserving Image Registration Model and Its Fast Solution

In §3.4 we remarked that the commonly used regularisers $\mathcal{R}^{\text{elas}}$, $\mathcal{R}^{\text{diff}}$, $\mathcal{R}^{\text{FMcurv}}$, and $\mathcal{R}^{\beta\text{TV}}$ yield $\boldsymbol{u}$ to be either global or piecewise smooth over the image domain. In this chapter we first present a variational model based on a modified TV regularisation with the so-called *potential function*, which can be interpreted as a half way model between diffusion (smooth) and TV (non-smooth) registration. The idea stems from image restoration, image reconstruction, and optical flow computation, where smoothing and preserving discontinuities of solutions are both important [4, 2, 3, 10, 20, 17, 18, 28, 40, 123]. Second to solve the resulting Euler-Lagrange system of two coupled, nonlinear PDEs, we present a multilevel strategy and an adaptive parameter selection procedure similar to the ones seen in Chapter 5. Numerical tests presented in this chapter using both synthetic and realistic images not only confirm that the proposed model is more robust in registration quality for a wide range of applications than previous models, but also that the proposed multilevel approach can deliver an acceptable solution many orders of magnitude faster than the gradient descent approach, popularly used in image processing.

6.1 Introduction

Let R and T denote a reference and a template image, respectively. Here the given images R and T are modelled as the continuous functions mapping from an image domain $\Omega = [0,1]^2 \subset \mathbb{R}^2$ into $V = [0,1] \subset \mathbb{R}_0^+$. The registration aims at finding a reasonable deformation field $\boldsymbol{u} = (u_1, u_2)^\top : \mathbb{R}^2 \to \mathbb{R}^2$, whose components u_1 and u_2 are functions of the variable $\mathbf{x} = (x_1, x_2)^\top$ in the image domain Ω, such that the transformed version of the template image $T_{\boldsymbol{u}} = T_{\boldsymbol{u}}(\mathbf{x}) = T(\mathbf{x} + \boldsymbol{u}(\mathbf{x}))$ becomes similar to R in a geometrical sense. As already pointed out in §3.2 the registration problem can be posed as minimisation of the functional:

$$\mathcal{J}_\alpha(\boldsymbol{u}) = \mathcal{D}^{\text{SSD}}(R, T_{\boldsymbol{u}}) + \alpha \mathcal{R}(\boldsymbol{u}) \tag{6.1}$$

where the image intensities of the given images R and T are assumed to be comparable and $\alpha > 0$ is the regularisation parameter. Recall that

$$\mathcal{D}^{\mathrm{SSD}}(\boldsymbol{u}) = \frac{1}{2}\int_{\Omega}\left(T(\mathbf{x}+\boldsymbol{u}(\mathbf{x})) - R(\mathbf{x})\right)^2 d\mathbf{x}, \tag{6.2}$$

and

$$\boldsymbol{f}(\boldsymbol{u}) = (f_1(\boldsymbol{u}), f_2(\boldsymbol{u}))^{\top} = ((T_{\boldsymbol{u}} - R)\,\partial_{u_1}T_{\boldsymbol{u}}, (T_{\boldsymbol{u}} - R)\,\partial_{u_2}T_{\boldsymbol{u}})^{\top} \tag{6.3}$$

is related to the first variation of $\mathcal{D}^{\mathrm{SSD}}$.

The rest of the chapter is organized as follows. §6.2 introduces a discontinuity-preserving image registration model. §6.3 discusses the numerical solution methods for the resulting Euler-Lagrange system. §6.4 presents our multilevel approach based on a FAS-NMG method. Experimental results from synthetic and realistic images are illustrated in §6.6, followed by conclusions in §6.7.

6.2 A discontinuity-preserving image registration model

Motivated by several regularisation techniques that have proved to be very useful in optical flow computation [2, 10, 20, 17, 40], image reconstruction [28], and image restoration [4, 3, 9, 123], one can smooth isotropically each component of $\boldsymbol{u}$ inside homogeneous (or flat) regions corresponding to weak gradients and preserve its discontinuities in inhomogeneous regions presenting large gradients by replacing $|\nabla z|$ in (3.37) (where z is u_1 or u_2) by the so-called *potential function* (or *Lorentzian error function used in statistics*) $\phi(|\nabla z|)$ satisfying some conditions to preserve discontinuities of z. Consequently, the modified TV (MTV) model can be represented in terms of a general ϕ (which exclude, $\phi(s) = s$, the choice of the TV)

$$\mathcal{R}^{\mathrm{MTV}}(\boldsymbol{u}) = \sum_{l=1}^{2}\int_{\Omega}\phi(|\nabla u_l|)\,d\mathbf{x}. \tag{6.4}$$

Due to the sum rule, the following theorem can be used to compute the first variation of $\mathcal{R}^{\mathrm{MTV}}$.

Theorem 6.2.1 *Let ϕ be a given function and let*

$$\overline{\mathcal{R}}^{MTV}(u_l) = \int_{\Omega}\phi(|\nabla u_l|)\,d\mathbf{x}.$$

Then the first variation of $\overline{\mathcal{R}}^{MTV}(u_l)$ is given by

$$\delta\overline{\mathcal{R}}^{MTV}(u_l;\eta_l) = -\int_{\Omega}(\nabla\cdot(\frac{\phi'(|\nabla u_l|)}{|\nabla u_l|}\nabla u_l))\eta_l d\mathbf{x} + \int_{\partial\Omega}\left\langle\frac{\phi'(|\nabla u_l|)}{|\nabla u_l|}\nabla u_l, \boldsymbol{n}\right\rangle_{\mathbb{R}^2}\eta_l d\mathbf{x}. \tag{6.5}$$

Proof.

$$
\begin{aligned}
\delta\overline{\mathcal{R}}^{\mathrm{MTV}}(u_l;\eta_l) &= \frac{d}{d\epsilon}\overline{\mathcal{R}}^{\mathrm{MTV}}(u_l+\epsilon\eta_l)\Big|_{\epsilon=0} \\
&= \frac{d}{d\epsilon}\int_\Omega \phi(|\nabla(u_l+\epsilon\eta_l)|)d\mathbf{x}\Big|_{\epsilon=0} \\
&= \int_\Omega \frac{d}{d\epsilon}\phi(|\nabla(u_l+\epsilon\eta_l)|)\Big|_{\epsilon=0} d\mathbf{x} \\
&= \int_\Omega \sum_{m=1}^{2} \phi'(\sqrt{[(u_{l_{x_1}}+\epsilon\eta_{lx_1})^2+(u_{l_{x_2}}+\epsilon\eta_{lx_2})^2]})\times \\
&\quad \frac{\partial\sqrt{[(u_{l_{x_1}}+\epsilon\eta_{lx_1})^2+(u_{l_{x_2}}+\epsilon\eta_{lx_2})^2]}}{\partial(u_{l_{x_m}}+\epsilon\eta_{lx_m})}\times \\
&\quad \frac{\partial(u_{l_{x_m}}+\epsilon\eta_{lx_m})^2}{\partial(u_{l_{x_m}}+\epsilon\eta_{lx_m})}\frac{\partial(u_{l_{x_m}}+\epsilon\eta_{lx_m})}{\partial\epsilon}\Big|_{\epsilon=0} d\mathbf{x} \\
&= \int_\Omega \sum_{m=1}^{2} \frac{\phi'(|\nabla u_l|)u_{l_{x_m}}}{|\nabla u_l|}\eta_{lx_m} d\mathbf{x} \\
&= \int_\Omega \left\langle \phi'(|\nabla u_l|)\frac{\nabla u_l}{|\nabla u_l|}, \nabla\eta_l \right\rangle_{\mathbb{R}^2} d\mathbf{x}.
\end{aligned}
$$

After applying the divergence theorem, we have

$$
\delta\overline{\mathcal{R}}^{\mathrm{MTV}}(u_l;\eta_l) = -\int_\Omega (\nabla\cdot(\frac{\phi'(|\nabla u_l|)}{|\nabla u_l|}\nabla u_l))\eta_l d\mathbf{x} + \int_{\partial\Omega}\left\langle \phi'(|\nabla u_l|)\frac{\nabla u_l}{|\nabla u_l|}, \boldsymbol{n}\right\rangle_{\mathbb{R}^2}\eta_l d\boldsymbol{s},
$$

which concludes the proof. ■

Let $\mathcal{R}(\boldsymbol{u}) = \mathcal{R}^{\mathrm{MTV}}(\boldsymbol{u})$. Then by the sum rule, (3.8), and Theorem 6.2.1, the Euler-Lagrange equations of (6.1) are given by

$$
\begin{cases} f_1(\boldsymbol{u}) - \alpha\nabla\cdot(\frac{\phi'(|\nabla u_1|)}{|\nabla u_1|}\nabla u_1) = 0 \\ f_2(\boldsymbol{u}) - \alpha\nabla\cdot(\frac{\phi'(|\nabla u_2|)}{|\nabla u_2|}\nabla u_2) = 0 \end{cases}, \qquad \text{(MTV)}. \tag{6.6}
$$

subject to the natural boundary condition $\partial_{\boldsymbol{n}} u_l = 0$ on $\partial\Omega$ for $l = 1, 2$.

We remark first that the boundary condition $\partial_{\boldsymbol{n}} u_l = 0$ on $\partial\Omega$ is used to drop the boundary integral in (6.5). Second, there exist many choices for the potential function ϕ. In order to define and identify the potential function ϕ [4, 2, 3, 10, 9, 20, 17, 28, 40, 123] for modifying the TV model, below, we give some commonly used ones for (6.6) and its diffusion coefficient $D(s) = \phi'(s)/s$ ($s = |\nabla z|$, $z = u_1$ or u_2):

- $\phi(s) = \frac{1}{p}s^p$, $D(s) = \frac{1}{s^{2-p}}$, $1 < p < 2$ (this ϕ is related to (3.37) when $p = 1$)
- $\phi(s) = \log(1+s^2)$, $D(s) = \frac{2}{1+s^2}$ (Perona-Molik's model)
- $\phi(s) = \frac{s^2}{1+s^2}$, $D(s) = \frac{2}{(1+s^2)^2}$ (Geman-Reynolds's model)
- $\phi(s) = 2\sqrt{1+s^2} - 2$, $D(s) = \frac{2}{\sqrt{1+s^2}}$ (Aubert's model)
- $\phi(s) = 2\log[\cosh(s)]$, $D(s) = \begin{cases} 2 & , s = 0 \\ 2\tanh(s)/s & , s \neq 0 \end{cases}$ (Green's model)

Note that the diffusion coefficient (or the stopping function) $D(s)$ has the following basic properties: (1) $D(s) \to 0$ as $s \to \infty$. (2) $D(s) \to M$ $(0 < M < +\infty)$ as $s \to 0$. These mean that on one hand it preserves discontinuities of $\boldsymbol{u}$ by reducing or stopping the diffusion (smoothing) process in inhomogeneous regions, on the other hand it smooths $\boldsymbol{u}$ isotropically inside homogeneous regions. In other words, TV-like regularisation is used in inhomogeneous regions and diffusion- or quadratic-like regularisation is used in homogeneous regions.

In this study, we focus only on Perona-Molik's model defined by $\phi(s) = \log(1 + s^2)$ for this kind of regularisation techniques, and then (6.6) becomes

$$\begin{cases} \mathcal{N}_1(\boldsymbol{u}) = f_1(\boldsymbol{u}) \overbrace{-\alpha\nabla \cdot \left(\frac{2\nabla u_1}{1 + (|\nabla u_1|)^2} \right)}^{\mathcal{L}_{\mathrm{NL}}(u_1)} = g_1 = 0, \\ \mathcal{N}_2(\boldsymbol{u}) = f_2(\boldsymbol{u}) \overbrace{-\alpha\nabla \cdot \left(\frac{2\nabla u_2}{1 + (|\nabla u_2|)^2} \right)}^{\mathcal{L}_{\mathrm{NL}}(u_2)} = g_2 = 0 \end{cases} \tag{6.7}$$

subject to $\partial_{\boldsymbol{n}} u_1 = \partial_{\boldsymbol{n}} u_2 = 0$ on $\partial\Omega$. Here $\mathcal{N}_l$ and $\mathcal{L}_{\mathrm{NL}}(u_l)$ are the nonlinear partial differential operators (for $l = 1, 2$), $\boldsymbol{n} = (n_1, n_2)^\top$ is the outward unit vector normal to the image boundary $\partial\Omega$, and $g_l = 0$ is technical notation for numerical solutions that will be used in the coming sections. Note that other choices of ϕ can be considered in the similar way.

Remark 6.2.1 *Although the regularisation technique* (6.4) *is not completely new for other image processing applications, to best of our knowledge it is new for deformable image registration based on the nonlinear fitting term* $\mathcal{D}^{SSD}$ *(6.2). Particularly, the nonlinear data term* $f_l(\boldsymbol{u}) = (T_{\boldsymbol{u}} - R)\,\partial_{u_l} T_{\boldsymbol{u}}$ *in* (6.7) *is totally different from those of linear ones used in image reconstruction, image restoration, and optical flow models; c.f. [4, 2, 3, 10, 9, 20, 17, 28, 40, 123]. Therefore, an effective technique of* (6.7) *is much more challenging.*

6.3 Numerical solutions of the PDE system (6.7)

The section will be started firstly by discretising the Euler-Lagrange equations (6.7), followed by a discussion of numerical solutions for the discrete system.

6.3.1 Finite difference discretisation

Let the discrete domain

$$\Omega_h = \{\mathbf{x} \in \Omega | \mathbf{x} = (x_{1_i}, x_{2_j})^\top = ((2i-1)\,h_1/2, (2j-1)\,h_2/2)^\top,\ 1 \le i \le n_1, \quad 1 \le j \le n_2$$

consist of $n = n_1 n_2$ cells of size $h_1 \times h_2$ with grid spacing $h = (h_1, h_2) = (1/n_1, 1/n_2)$ and let $(u_l^h)_{i,j} = u_l^h(x_{1_i}, x_{2_j})$ denote the grid functions for $l = 1, 2$. Applying finite difference schemes based on the cell-centered grid points to discretise (6.7), the discrete system at a grid point

(i,j) is given by

$$\begin{cases} \mathcal{N}_1^h(\boldsymbol{u}^h)_{i,j} = f_1^h(u_1^h, u_2^h)_{i,j} + \alpha \mathcal{L}_{\mathrm{NL}}^h(u_1^h)_{i,j} = (g_1^h)_{i,j} \\ \mathcal{N}_2^h(\boldsymbol{u}^h)_{i,j} = f_2^h(u_1^h, u_2^h)_{i,j} + \alpha \mathcal{L}_{\mathrm{NL}}^h(u_2^h)_{i,j} = (g_2^h)_{i,j} \end{cases} \tag{6.8}$$

subject to the discrete Neumann boundary conditions,

$$(u_l^h)_{i,1} = (u_l^h)_{i,2}, \quad (u_l^h)_{i,n_2} = (u_l^h)_{i,n_2-1}, \quad \left(u_l^h\right)_{1,j} = (u_l^h)_{2,j}, \quad (u_l^h)_{n_1,j} = (u_l^h)_{n_1-1,j}, \tag{6.9}$$

with the following notation

$$\begin{aligned}
\mathcal{L}_{\mathrm{NL}}\left(u_l^h\right)_{i,j} &= -\left[\frac{\delta_{x_1}^-}{h_1}\left(\frac{2\delta_{x_1}^+(u_l^h)_{i,j}/h_1}{1+(\delta_{x_1}^+(u_l^h)_{i,j}/h_1)^2+(\delta_{x_2}^+(u_l^h)_{i,j}/h_2)^2}\right)\right. \\
&\quad \left. + \frac{\delta_{x_2}^-}{h_2}\left(\frac{2\delta_{x_2}^+(u_l)_{i,j}/h_2}{1+(\delta_{x_1}^+(u_l^h)_{i,j}/h_1)^2+(\delta_{x_2}^+(u_l^h)_{i,j}/h_2)^2}\right)\right] \\
&= (\Sigma_l^h)_{i,j}(u_l^h)_{i,j} - (\overline{\Sigma}_l^h)_{i,j}(u_l^h)_{i,j} \\
\left(\Sigma_l^h\right)_{i,j} &= \left(1+\gamma^2\right) D_{l3}[(u_l^h)_{i,j}] + D_{l1}[(u_l^h)_{i,j}] + \gamma^2 D_{l2}[(u_l^h)_{i,j}], \quad \gamma = h_1/h_2, \\
(\overline{\Sigma}_l^h)_{i,j}(u_l^h)_{i,j} &= D_{l3}[(u_l^h)_{i,j}]((u_l^h)_{i+1,j} + \gamma^2(u_l^h)_{i,j+1}) \\
&\quad + D_{l1}[(u_l^h)_{i,j}](u_l^h)_{i-1,j} + \gamma^2 D_{l2}[(u_l^h)_{i,j}](u_l^h)_{i,j-1}), \\
D_{l1}[(u_l^h)_{i,j}] &= D[(u_l^h)_{i-1,j}], \quad D_{l2}[(u_l^h)_{i,j}] = D[(u_l^h)_{i,j-1}], \quad D_{l3}[(u_l^h)_{i,j}] = D[(u_l^h)_{i,j}] \\
D[\left(u_l^h\right)_{i,j}] &= 2/\left(h_1^2 + (\delta_{x_1}^+(u_l^h)_{i,j})^2 + (\gamma\delta_{x_2}^+(u_l^h)_{i,j})^2\right), \quad 1 \le i \le n_1, \quad 1 \le j \le n_2, \\
\delta_{x_1}^{\pm}\left(u_l^h\right)_{i,j} &= \pm\left((u_l^h)_{i\pm1,j} - (u_l^h)_{i,j}\right), \\
\delta_{x_2}^{\pm}\left(u_l^h\right)_{i,j} &= \pm\left((u_l^h)_{i,j\pm1} - (u_l^h)_{i,j}\right), \\
f_1^h\left(u_1^h, u_2^h\right)_{i,j} &= (I_{i,j}^{h^*} - R_{i,j}^h)((I_{i+1,j}^{h^*} - I_{i-1,j}^{h^*})/(2h_1)), \\
f_2^h\left(u_1^h, u_2^h\right)_{i,j} &= (I_{i,j}^{h^*} - R_{i,j}^h)((I_{i,j+1}^{h^*} - I_{i,j-1}^{h^*})/(2h_2)), \\
I_{i,j}^{h^*} &= I^h(i + \left(u_1^h\right)_{i,j}, j + \left(u_2^h\right)_{i,j}), \\
(\boldsymbol{u}^h)_{i,j} &= ((u_1^h)_{i,j}, (u_2^h)_{i,j})^\top.
\end{aligned}$$

Here $(g_1^h)_{i,j} = (g_2^h)_{i,j} = 0$ on the finest grid in multigrid setting.

6.3.2 Method 1 – An explicit time marching (ETM) method

As mentioned in §3.5.1, a time marching scheme is one of convenient ways to solve the resulting Euler-Lagrange equations like (6.7). The main idea is to introduce an artificial time variable t and compute the steady-state solution of the system of time-dependent PDEs of the form:

$$\begin{cases} \partial_t u_1(\mathbf{x}, t) + \mathcal{N}_1(\boldsymbol{u}(\mathbf{x}, t)) = g_1(\mathbf{x}) \\ \partial_t u_2(\mathbf{x}, t) + \mathcal{N}_2(\boldsymbol{u}(\mathbf{x}, t)) = g_2(\mathbf{x}) \end{cases}.$$

In order to overcome the nonlinearity of $\mathcal{N}_l$, the so-called *explicit scheme* can be conveniently applied, and the iteration is then given by

$$\begin{cases} \partial_t u_1(\mathbf{x}, t_{k+1}) = g_1(\mathbf{x}) - \mathcal{N}_1(\boldsymbol{u}(\mathbf{x}, t_k)) \\ \partial_t u_2(\mathbf{x}, t_{k+1}) = g_2(\mathbf{x}) - \mathcal{N}_2(\boldsymbol{u}(\mathbf{x}, t_k)) \end{cases} \qquad k = 0, 1, 2, 3, \ldots$$

where $\boldsymbol{u}(\mathbf{x}, t_0)$ is some initial displacement fields, typically $\boldsymbol{u}(\mathbf{x}, t_0) = 0$.

For the time discretisation we introduce a time-step $\tau > 0$, and then $\boldsymbol{u}$ is updated at the time step $k+1$ by

$$\begin{cases} u_1(\mathbf{x}, t_{k+1}) = u_1(\mathbf{x}, t_k) + \tau\left[g_1(\mathbf{x}) - \mathcal{N}_1(\boldsymbol{u}(\mathbf{x}, t_k))\right] \\ u_2(\mathbf{x}, t_{k+1}) = u_2(\mathbf{x}, t_k) + \tau\left[g_2(\mathbf{x}) - \mathcal{N}_2(\boldsymbol{u}(\mathbf{x}, t_k))\right] \end{cases}$$

which we simply denote by

$$\begin{cases} (u_1^{(k+1)})_{i,j} = (u_1^{(k)})_{i,j} + \tau[g_{1_{i,j}} - \mathcal{N}_1(\boldsymbol{u}^{(k)})_{i,j}] \\ (u_2^{(k+1)})_{i,j} = (u_2^{(k)})_{i,j} + \tau[g_{2_{i,j}} - \mathcal{N}_2(\boldsymbol{u}^{(k)})_{i,j}] \end{cases} \tag{6.10}$$

We note that this time-marching scheme is easy to implement, but very slow to converge because the length of the time-step τ is required to be a very small number for stability reasons.

6.3.3 Method 2 – A semi-implicit time marching (SITM) method

In order to speed up the convergence of (6.10), we may apply the fully implicit scheme, and then $\boldsymbol{u}^{(k+1)}$ is updated by

$$\begin{cases} (u_1^{(k+1)})_{i,j} = (u_1^{(k)})_{i,j} + \tau[g_{1_{i,j}} - \mathcal{N}_1(\boldsymbol{u}^{(k+1)})_{i,j}] \\ (u_2^{(k+1)})_{i,j} = (u_2^{(k)})_{i,j} + \tau[g_{2_{i,j}} - \mathcal{N}_2(\boldsymbol{u}^{(k+1)})_{i,j}] \end{cases}, \qquad k = 0, 1, 2, 3, \ldots \tag{6.11}$$

In order to cope with the nonlinearity of $\mathcal{N}_l$, we may linearise (6.11) respect to the $k+1$th time-step using the method of 'frozen coefficients' as well known for variational approaches related to the TV operator (see e.g. [6, 13, 22, 24, 31, 52, 122, 123]), and obtain the semi-implicit scheme as given by the following system of linear elliptic PDEs:

$$\begin{cases} (u_1^{(k+1)})_{i,j} + \tau\alpha\mathcal{L}_{\mathrm{NL}}^{\mathrm{lin}}[(u_1^{(k)})_{i,j}](u_1^{(k+1)})_{i,j} = (u_1^{(k)})_{i,j} + \tau[g_{1_{i,j}} - f_1(u_1^{(k)}, u_2^{(k)})_{i,j}] \\ (u_2^{(k+1)})_{i,j} + \tau\alpha\mathcal{L}_{\mathrm{NL}}^{\mathrm{lin}}[(u_2^{(k)})_{i,j}](u_2^{(k+1)})_{i,j} = (u_2^{(k)})_{i,j} + \tau[g_{2_{i,j}} - f_2(u_1^{(k)}, u_2^{(k)})_{i,j}] \end{cases} \tag{6.12}$$

where

$$\mathcal{L}_{\mathrm{NL}}^{\mathrm{lin}}[(u_l^{(k)})_{i,j}](u_l^{(k+1)})_{i,j} = (\Sigma_l^{(k)})_{i,j}(u_l^{(k+1)})_{i,j} - (\overline{\Sigma_l^{(k)}})_{i,j}(u_l^{(k+1)})_{i,j}. \tag{6.13}$$

Therefore, the update formula determined by a lexicographical ordering in a matrix vector form can be written as

$$\begin{cases} u_1^{(k+1)} = (\mathbf{I} + \tau\alpha\mathcal{L}_{\mathrm{NL}}^{\mathrm{lin}}[u_l^{(k)}])^{-1}(u_1^{(k)} + \tau g_1 - \tau f_1(u_1^{(k)}, u_2^{(k)})) \\ u_2^{(k+1)} = (\mathbf{I} + \tau\alpha\mathcal{L}_{\mathrm{NL}}^{\mathrm{lin}}[u_2^{(k)}])^{-1}(u_2^{(k)} + \tau g_2 - \tau f_2(u_1^{(k)}, u_2^{(k)})) \end{cases}, \tag{6.14}$$

where $\mathbf{I}$ is the identity matrix.

Since $\mathcal{L}_{\mathrm{NL}}^{\mathrm{lin}}[u_l^{(k)}]$ can be divided into two parts, (6.14) can be simplified further as

$$\begin{cases} u_1^{(k+1)} = (\mathbf{I} + \tau\alpha\sum_{m=1}^{2}\mathcal{L}_{\mathrm{NL}}^{\mathrm{lin}_{x_m}}[u_1^{(k)}])^{-1}(u_1^{(k)} + \tau g_1 - \tau f_1(u_1^{(k)}, u_2^{(k)})) \\ u_2^{(k+1)} = (\mathbf{I} + \tau\alpha\sum_{m=1}^{2}\mathcal{L}_{\mathrm{NL}}^{\mathrm{lin}_{x_m}}[u_2^{(k)}])^{-1}(u_2^{(k)} + \tau g_2 - \tau f_2(u_1^{(k)}, u_2^{(k)})) \end{cases}. \tag{6.15}$$

where $\mathcal{L}_{\mathrm{NL}}^{\mathrm{lin}}[u_l^{(k)}] = \mathcal{L}_{\mathrm{NL}}^{\mathrm{lin}_{x_1}}[u_l^{(k)}] + \mathcal{L}_{\mathrm{NL}}^{\mathrm{lin}_{x_2}}[u_l^{(k)}]$, and $\mathcal{L}_{\mathrm{NL}}^{\mathrm{lin}_{x_m}}[u_l^{(k)}]$ is a finite difference approximation of the second-order derivative of $u_l^{(k)}$ with respect to the mth coordinate (for $m = 1, 2$).

6.3.4 Method 3 – An additive operator splitting (AOS) method

The AOS scheme [46, 93, 138] is more efficient than the standard semi-implicit scheme (6.15). The basic idea is to replace the inverse of the sum by a sum of inverses. The corresponding iterates are then defined by

$$\begin{cases} u_1^{(k+1)} = \frac{1}{2}\sum_{m=1}^{2}(\mathbf{I} + 2\tau\alpha\mathcal{L}_{\mathrm{NL}}^{\mathrm{lin}x_m}[u_1^{(k)}])^{-1}(u_1^{(k)} + \tau g_1 - \tau f_1(u_1^{(k)}, u_2^{(k)})) \\ u_2^{(k+1)} = \frac{1}{2}\sum_{m=1}^{2}(\mathbf{I} + 2\tau\alpha\mathcal{L}_{\mathrm{NL}}^{\mathrm{lin}x_m}[u_2^{(k)}])^{-1}(u_2^{(k)} + \tau g_2 - \tau f_2(u_1^{(k)}, u_2^{(k)})) \end{cases}. \tag{6.16}$$

which is much cheaper than those obtained from (6.15) because the two diagonal systems in each component are solved per iteration rather than the 5-band system.

Remark 6.3.1 *Although each linear elliptic PDE given by (6.12) is solved by a fast solution method (e.g. a linear multigrid technique), the number of time steps k in fulfilling the necessary condition for being a minimiser of the variational problem represented by (6.1), i.e. in achieving a kind of convergence, may not be small; see Table 6.2. The reason is that the gradient descent technique requires to solve the linear system many times with changing the right-hand side of (6.12). It is reasonable to develop a new and fast solution method in solving directly the Euler-Lagrange equations (6.7), which will be explained in the Method 4 and §6.4.*

6.3.5 Method 4 – A stabilised fixed-point (SFP) method

As is well-known fixed-point (FP) methods are more robust than those of time marching techniques when appropriate FP schemes are applied, especially for problems related to the TV operator like (6.7). Due to Neumann boundary conditions, the standard FP scheme of the discrete system (6.8) given by

$$\begin{cases} \alpha\mathcal{L}_{\mathrm{NL}}^{\mathrm{lin}}[u_1^{[\nu]}]u_1^{[\nu+1]} = g_1 - f_1(u_1^{[\nu]}, u_2^{[\nu]}) = G_1[\boldsymbol{u}^{[\nu]}] \\ \alpha\mathcal{L}_{\mathrm{NL}}^{\mathrm{lin}}[u_2^{[\nu]}]u_2^{[\nu+1]} = g_2 - f_2(u_1^{[\nu]}, u_2^{[\nu]}) = G_2[\boldsymbol{u}^{[\nu]}] \end{cases} \tag{6.17}$$

leads to the singular problem for each FP (or outer iteration step) ν ($\nu = 0, 1, 2, ...$) and then a special treatment is required; see §5.2 – 5.3 for the similar problem occurred in the discrete system resulting from the diffusion model. Here the frozen operator

$$\mathcal{L}_{\mathrm{NL}}^{\mathrm{lin}}[(u_l^{[\nu]})_{i,j}]u_l^{[\nu+1]} = (\Sigma_l^{[\nu]})_{i,j}(u_l^{[\nu+1]})_{i,j} - (\overline{\Sigma}_l^{[\nu]})_{i,j}(u_l^{[\nu+1]})_{i,j}$$

by the so-called *Lagged-diffusivity* method [26] or Quasi-Newton scheme [137] is used and the symbol h and $(\cdot)_{i,j}$ in (6.17) are dropped for simplicity.

To stabilise the FP scheme (6.17), we again apply the linearisation idea of the data term given by

$$f_l(u_1^{[\nu+1]}, u_2^{[\nu+1]}) \sim f_l(u_1^{[\nu]}, u_2^{[\nu]}) + \sigma_{l1}^{[\nu]}(u_1^{[\nu+1]} - u_1^{[\nu]}) + \sigma_{l2}^{[\nu]}(u_2^{[\nu+1]} - u_2^{[\nu]}),$$

where

$$\sigma_{l1}(\boldsymbol{u}^{[\nu]}) = \partial_{u_1} f_l(u_1^{[\nu]}, u_2^{[\nu]}) = (\partial_{u_l} T_{\boldsymbol{u}^{[\nu]}})(\partial_{u_1} T_{\boldsymbol{u}^{[\nu]}}) + (T_{\boldsymbol{u}^{[\nu]}} - R)(\partial_{u_1 u_l} T_{\boldsymbol{u}^{[\nu]}}),$$

and

$$\sigma_{l2}(\boldsymbol{u}^{[\nu]}) = \partial_{u_2} f_l(u_1^{[\nu]}, u_2^{[\nu]}) = (\partial_{u_l} T_{\boldsymbol{u}^{[\nu]}})(\partial_{u_2} T_{\boldsymbol{u}^{[\nu]}}) + (T_{\boldsymbol{u}^{[\nu]}} - R)(\partial_{u_2 u_l} T_{\boldsymbol{u}^{[\nu]}}).$$

This yields the linearised system

$$\mathbf{N}[\boldsymbol{u}^{[\nu]}]\boldsymbol{u}^{[\nu+1]} = \mathbf{G}[\boldsymbol{u}^{[\nu]}], \tag{6.18}$$

where $\boldsymbol{u}^{[\nu+1]} = (u_1^{[\nu+1]}, u_2^{[\nu+1]})^\top$,

$$\mathbf{N}[\boldsymbol{u}^{[\nu]}] = \begin{bmatrix} \mathcal{L}_{\mathrm{NL}}^{\mathrm{lin}}[u_1^{[\nu]}] + \sigma_{11}(\boldsymbol{u}^{[\nu]}) & \sigma_{12}(\boldsymbol{u}^{[\nu]}) \\ \sigma_{21}(\boldsymbol{u}^{[\nu]}) & \mathcal{L}_{\mathrm{NL}}^{\mathrm{lin}}[u_2^{[\nu]}] + \sigma_{22}(\boldsymbol{u}^{[\nu]}) \end{bmatrix},$$

and

$$\mathbf{G}[\boldsymbol{u}^{[\nu]}] = \begin{pmatrix} G_1[\boldsymbol{u}^{[\nu]}] + \sigma_{11}(\boldsymbol{u}^{[\nu]})u_1^{[\nu]} + \sigma_{12}(\boldsymbol{u}^{[\nu]})u_2^{[\nu]} \\ G_2[\boldsymbol{u}^{[\nu]}] + \sigma_{21}(\boldsymbol{u}^{[\nu]})u_2^{[\nu]} + \sigma_{21}(\boldsymbol{u}^{[\nu]})u_1^{[\nu]} \end{pmatrix}.$$

As mentioned in §5.3.1, $\sigma_{21}^{[\nu]} = \sigma_{12}^{[\nu]}$ and $\sigma_{lm}^{[\nu]} = (\partial_{u_l} T_{\boldsymbol{u}^{[\nu]}})(\partial_{u_m} T_{\boldsymbol{u}^{[\nu]}})$ for $m = 1, 2$ is used in our numerical scheme. We shall call (6.18) the *stabilised fixed-point (SFP) method.*

As a common way to solve (6.18) for each FP or outer step ν, we use the $\omega-$PCGS relaxation method with the relaxation parameter $\omega \in (0, 2)$ and then its new step is given by

$$(\boldsymbol{u}^{[\nu+1]})_{i,j}^{[k+1]} = (1-\omega)\,(\boldsymbol{u}^{[\nu+1]})_{i,j}^{[k]} + \omega(\mathbf{N}[\boldsymbol{u}^{[\nu]}]_{i,j})^{-1}(\mathbf{G}[\boldsymbol{u}^{[\nu]}])_{i,j}^{[k+1/2]}, \tag{6.19}$$

where

$$\mathbf{N}[\boldsymbol{u}^{[\nu]}]_{i,j} = \begin{bmatrix} (\Sigma_1^{[\nu]})_{i,j} + (\sigma_{11}(\boldsymbol{u}^{[\nu]}))_{i,j} & (\sigma_{12}(\boldsymbol{u}^{[\nu]}))_{i,j} \\ (\sigma_{21}(\boldsymbol{u}^{[\nu]}))_{i,j} & (\Sigma_2^{[\nu]})_{i,j} + (\sigma_{22}(\boldsymbol{u}^{[\nu]}))_{i,j} \end{bmatrix},$$

$$(\mathbf{G}[\boldsymbol{u}^{[\nu]}])_{i,j}^{[k+1/2]} = \begin{pmatrix} (G_1[\boldsymbol{u}^{[\nu]}])_{i,j} + (\sigma_{11}(\boldsymbol{u}^{[\nu]}))_{i,j}(u_1^{[\nu]})_{i,j} + (\sigma_{12}(\boldsymbol{u}^{[\nu]}))_{i,j}(u_2^{[\nu]})_{i,j} \\ +\alpha(\overline{\Sigma}_1^{[\nu]})_{i,j}(u_1^{[\nu+1]})_{i,j}^{[k+1/2]} \\ \\ (G_2[\boldsymbol{u}^{[\nu]}])_{i,j} + (\sigma_{21}(\boldsymbol{u}^{[\nu]}))_{i,j}(u_1^{[\nu]})_{i,j} + (\sigma_{22}(\boldsymbol{u}^{[\nu]}))_{i,j}(u_2^{[\nu]})_{i,j} \\ +\alpha(\overline{\Sigma}_2^{[\nu]})_{i,j}(u_2^{[\nu+1]})_{i,j}^{[k+1/2]} \end{pmatrix},$$

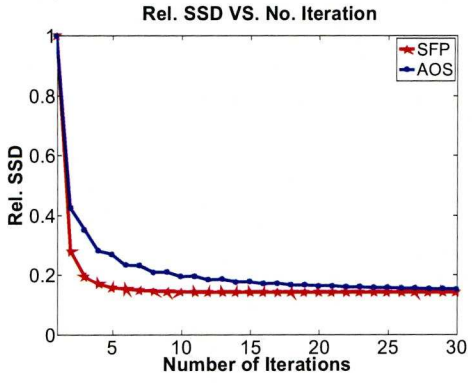
and

$$\begin{aligned}(\overline{\Sigma}_l^{[\nu]})_{i,j}(u_l^{[\nu+1]})_{i,j}^{[k+1/2]} &= D_{l3}[(u_l^{[\nu]})_{i,j}]((u_l^{[\nu+1]})_{i+1,j}^{[k]} + \gamma^2(u_l^{[\nu+1]})_{i,j+1}^{[k]}) \\ &\quad + D_{l1}[(u_l^{[\nu]})_{i,j}](u_l^{[\nu+1]})_{i-1,j}^{[k+1]} + \gamma^2 D_{l2}[(u_l^{[\nu]})_{i,j}](u_l^{[\nu+1]})_{i,j-1}^{[k+1]}).\end{aligned}$$

Similarly, as remarked in §5.3.1, the SFP method (6.18) shows the interaction between the actual FP or outer iteration that overcomes the nonlinearity of the operator $\mathcal{N}_l$ at each outer step ν and the $\omega-$PCGS method that solves the resulting linear system of equations at each corresponding inner step k. Instead of solving the linearised system (6.18) with very high precision, the $\omega-$PCGS method or inner iteration can perform only a *few iterations* to obtain an approximate solution at each outer step ν. This is likely the so-called *inexact lagged-diffusivity* method which have been widely used for solving other problems in image processing applications related to the TV operator (see e.g. [6, 13, 26, 22, 24, 31, 122, 123]). This procedure leads

to a slight difference of convergence in the FP scheme when it is used as a stand-alone solver, whereas the computational costs significantly reduce. Moreover, the relaxation parameter ω has a strong influence on the convergence speed. We usually use $\omega > 1$, typically $\omega = 1.85$, for both smooth and non-smooth registration problems because it results in speeding up the convergence by many orders of magnitude compared with the GS approach ($\omega = 1$). We also note that line relaxation techniques, e.g. alternative line relaxation, are optional for the inner step. However, they usually require more computational costs than those of the ω−PCGS method.

(a)Example 2 - Rel. SSD by 2 methods



(b)Example 2 - Rel. Res by 2 methods

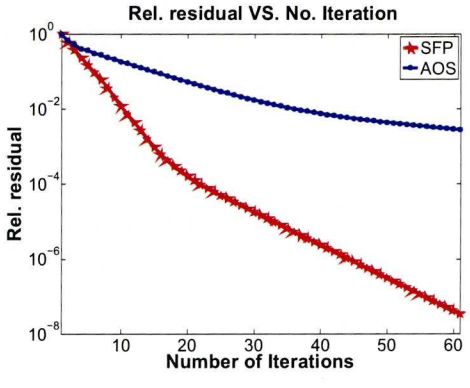


Figure 6.1: Numerical results by Method 3 (AOS (6.16)) and Method 4 (SFP) for Example 2 (in a 32×32 grid as shown in Figure 6.4 $(a) - (b)$) with $\tau = 0.05$, $\alpha = 0.1$, and $GSiter = 5$. (a) shows the relative errors in SSD and (b) shows the relative residuals versus iterations. Clearly Method 4 (SFP) performs much better than Method 3 (AOS).

Finally, the SFP method (6.18) on the fine grid can be summarised as follows:

Algorithm 6.3.1 (Algorithm for the SFP method)

We use these to be smoothing parameters:

α regularisation parameter

ω relaxation parameter

$GSiter$ the maximum number of ω−PCGS iterations

$$\left[w_1^h, w_2^h\right] \leftarrow Smoother\left(w_1^h, w_2^h, g_1^h, g_2^h, R^h, T^h, \alpha, \omega, GSiter\right)$$

- Use input parameters to compute $(\sigma_{lm}(\boldsymbol{w}^h))_{i,j}$, $(\mathbf{G}(\boldsymbol{w}^h)_{i,j})$, and $(\mathbf{N}(\boldsymbol{w}^h)_{i,j})^{-1}$ for $l, m = 1, 2$, $1 \le i \le n_1$, and $1 \le j \le n_2$ (Here $(\boldsymbol{w}^h)_{i,j} = ((w_1^h)_{i,j}, (w_2^h)_{i,j})^\top$)
- Perform ω−PCGS steps
  - for $k = 1 : GSiter$
    - for $i = 1 : n_1$
      - for $j = 1 : n_2$
        - Compute $(\boldsymbol{w}^h)_{i,j}^{(k+1)} = ((w_1^h)_{i,j}^{(k+1)}, (w_2^h)_{i,j}^{(k+1)})^\top)$ using (6.19)
      - end
    - end
  - end

We have so far presented four numerical methods for solving (6.7) where Method 2 is enforced by Method 3 and Method 1 is less efficient. So it remains to test the overall performances of the two methods (i.e. Method 3 and 4). We tested them only for the smooth problem Example 2 as shown respectively in Figure 6.4 $(a)-(b)$ for a 32×32 grid. The results from this test in Figure 6.1 $(a)-(b)$ show that the Method 4 performs much better than Method 3 as expected. We remark the results for the non-smoother problem shown in Figure 6.4 $(a)-(b)$ are similar to those of the smooth problem.

Although Method 4 is recommended as a unilevel method, our next task is to use Method 4 as a potential smoother in the FAS-NMG framework to speed up the solving of (6.7) with the multilevel strategy.

6.4 A nonlinear multigrid method

Below we apply the FAS-NMG method to solve the coupled system of nonlinear PDEs,

$$\begin{cases} \mathcal{N}_1^h(\boldsymbol{u}^h) = g_1^h \\ \mathcal{N}_2^h(\boldsymbol{u}^h) = g_2^h \end{cases}$$

involving the nonlinear partial differential operator $\mathcal{N}_l^h\left(\boldsymbol{u}^h\right)$ $(l = 1, 2)$ given by (6.8).

In our multigrid method, Method 4 (SFP) given by (6.18) is used as the smoother. The averaging and bi-linear interpolation techniques are employed respectively as the restriction and interpolation operators between Ω_h and Ω_H, denoted by I_h^H and I_H^h. The DCA approach is performed to compute the coarse-grid operator $\mathcal{N}_l^h(\boldsymbol{u}^h)$ consisting of two parts: $f_l^h(u_1^h, u_2^h)$ and $\mathcal{L}_{\mathrm{NL}}^h(u_l^h)$. To solve (6.8) numerically, our FAS-NMG method is applied recursively down to the coarsest grid consisting of a small number of grid points, typically 4×4, and may be summarised as follows:

Algorithm 6.4.1 (FAS Nonlinear Multigrid Method)

We use these FAS multigrid parameters:

ν_1 pre-smoothing steps on each level

ν_2 post-smoothing steps on each level

μ the number of multigrid cycles on each level (μ= 1 for V−cycling and μ= 2 for W−cycling). Here we present the V−cycle with μ= 1.

α regularisation parameter

ω relaxation parameter

$GSiter$ the maximum number of ω−PCGS

$$\boldsymbol{w}^h \leftarrow FASNMG(\boldsymbol{w}^h, \alpha, \overrightarrow{\varepsilon})$$

- Select α, $\overrightarrow{\varepsilon} = (\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4)$ and initial guess solutions $\boldsymbol{w}^h_{initial} = (w^h_1, w^h_2)^\top$ on the finest grid
- Set $K = 0$, $\left(\boldsymbol{w}^h\right)^K = \boldsymbol{w}^h_{initial}$, $\widetilde{\varepsilon}_2 = \varepsilon_2 + 1$, $\widetilde{\varepsilon}_3 = \varepsilon_3 + 1$, and $\widetilde{\varepsilon}_4 = \varepsilon_4 + 1$
- While ($K < \varepsilon_1$ AND $\widetilde{\varepsilon}_2 > \varepsilon_2$ AND $\widetilde{\varepsilon}_3 > \varepsilon_3$ AND $\widetilde{\varepsilon}_4 > \varepsilon_4$)
  - $\left(\boldsymbol{w}^h\right)^{K+1} = [w^h_1, w^h_2] \leftarrow FASCYC(w^h_1, w^h_2, \mathcal{N}^h_1, \mathcal{N}^h_2, g^h_1, g^h_2, R^h, T^h, \nu_1, \nu_2, \alpha, \omega, GSiter)$
  - $\widetilde{\varepsilon}_2 = max\{\|g^h_l - \mathcal{N}^h_l((\boldsymbol{w}^h)^{K+1})\|_2 / \|g^h_l - \mathcal{N}^h_l(\boldsymbol{w}^h_{initial})\|_2,\ l = 1, 2\}$
  - $\widetilde{\varepsilon}_3 = \mathcal{D}^h(R^h, T^h_{(\boldsymbol{w}^h)^{K+1}}) / \mathcal{D}^h(R^h, T^h)$
    [Recall that $\mathcal{D}^h(R^h, T^h_{(\cdot)}) \sim \frac{h_1 h_2}{2} \left\| R^h, T^h_{(\cdot)} \right\|^2_2$]
  - $\widetilde{\varepsilon}_4 = \left| \mathcal{D}^h(R^h, T^h_{(\boldsymbol{w}^h)^{K+1}}) - \mathcal{D}^h(R^h, T^h_{(\boldsymbol{w}^h)^K}) \right|$
  - $K = K + 1$
- end

where

$$[w^h_1, w^h_2] \leftarrow FASCYC(w^h_1, w^h_2, \mathcal{N}^h_1, \mathcal{N}^h_2, g^h_1, g^h_2, R^h, T^h, \nu_1, \nu_2, \alpha, \omega, GSiter)$$

- If $\Omega_h = coarset\ grid$ ($|\Omega_h| = 4 \times 4$), solve (6.8) using time-marching techniques in §6.3.3 and then stop. Else continue with following step.
- Pre-smoothing:
  For $z = 1$ to ν_1, $\left[w^h_1, w^h_2\right] \leftarrow Smoother\left(w^h_1, w^h_2, g^h_1, g^h_2, R^h, T^h, \alpha, \omega, GSiter\right)$
- Restriction to the coarse grid:
  $w^H_1 \leftarrow I^H_h w^h_1, \qquad w^H_2 \leftarrow I^H_h w^h_2, \qquad R^H \leftarrow I^H_h R^h, \qquad T^H \leftarrow I^H_h T^h$
- Set the initial solution for the coarse-grid problem:
  $\left[\overline{w}^H_1, \overline{w}^H_2\right] \leftarrow \left[w^H_1, w^H_2\right]$
- Compute the new right-hand side for the coarse-grid problem:
  $g^H_1 \leftarrow I^H_h \left(g^h_1 - \mathcal{N}^h_1\left(w^h_1, w^h_2\right)\right) + \mathcal{N}^H_1\left(w^H_1, w^H_2\right)$
  $g^H_2 \leftarrow I^H_h \left(g^h_2 - \mathcal{N}^h_2\left(w^h_1, w^h_2\right)\right) + \mathcal{N}^H_2\left(w^H_1, w^H_2\right)$
- Implement the FAS multigrid on the coarse-grid problem:
  For $z = 1$ to μ,
  $\left[w^H_1, w^H_2\right] \leftarrow FASCYC\left(w^H_1, w^H_2, \mathcal{N}^H_1, \mathcal{N}^H_2, g^H_1, g^H_2, R^H, T^H, \nu_1, \nu_2, \alpha, \omega, GSiter\right)$
- Add the coarse-grid corrections:
  $w^h_1 \leftarrow w^h_1 + I^h_H\left(w^H_1 - \overline{w}^H_1\right), \qquad w^h_2 \leftarrow w^h_2 + I^h_H\left(w^H_2 - \overline{w}^H_2\right)$
- Post-smoothing:
  For $z = 1$ to ν_2, $\left[w^h_1, w^h_2\right] \leftarrow Smoother\left(w^h_1, w^h_2, g^h_1, g^h_2, R^h, T^h, \alpha, \omega, GSiter\right)$

For practical applications Algorithm 6.4.1 is stopped whenever the maximum number of V− or W−cycles ε_1 is reached (usually $\varepsilon_1 = 20$), the maximum value of the relative residuals obtained from the Euler-Lagrange equations is smaller than a small number $\varepsilon_2 > 0$ (typically $\varepsilon_2 = 10^{-8}$ for a convergent test and only $\varepsilon_2 = 10^{-2}$ for a practical application), the relative reduction of the dissimilarity is smaller than some $\varepsilon_3 > 0$ (we usually assign $\varepsilon_3 = 0.20$ meaning that the relative reduction of the dissimilarity would decrease about 80%), or the change in two consecutive steps of the data/fitting term $\mathcal{D}$ is smaller than a small number $\varepsilon_4 > 0$ (typically $\varepsilon_4 = 10^{-8}$).

6.5 A robust approach for discontinuity-preserving image registration (RADPIR)

As is well-known, we have to be carefully select α to provide well registered images because it is in general unknown a priori. To this end, we follow the *cooling* process presented in §5.3

and name this algorithm by *a robust approach for discontinuity-preserving image registration* (RADPIR), which can be summarised as follows:

Algorithm 6.5.1 (The basic RADPIR)

---

1. Input $\overrightarrow{\varepsilon}_{lo} = (2, 10^{-4}, 0.2, 10^{-4})$ and $\overrightarrow{\varepsilon}_{hi} = (20, 10^{-8}, 0.2, 10^{-8})$. Set $\alpha = 1$ (optional).
2. Obtain the optimal regularisation parameter α:
   - $[\boldsymbol{w}^{(0)}, \alpha] \leftarrow cooling(\boldsymbol{w}, \alpha, \overrightarrow{\varepsilon}_{lo})$.
3. Solve the discrete minimisation problem of (6.1) on the finest level using the found α:
   - $\boldsymbol{w} \leftarrow FASNMG(\boldsymbol{w}^{(0)}, \alpha, \overrightarrow{\varepsilon}_{hi})$

---

where

$$[\boldsymbol{w}^*, \alpha^*] \leftarrow cooling(\boldsymbol{w}, \alpha^{(0)}, \overrightarrow{\varepsilon})$$

- Set $s = 1$, $\boldsymbol{w}^{(s)} = \boldsymbol{w}^{(0)}$, $\alpha^{(s)} = \alpha^{(0)}$, $\eta = 0.5$.
- Outer iteration: For $s = 1, 2, 3, ...$
  - 1. Set $\alpha^{(s+1)} = \eta\alpha^{(s)}$ in $[5 \times 10^{-5}, \alpha^{(s)}]$
  - 2. Inner iteration: $\boldsymbol{w}_{new} \leftarrow FASNMG(\boldsymbol{w}^{(s)}, \alpha^{(s+1)}, \overrightarrow{\varepsilon})$
  - 3. If $\mathcal{J}_{\alpha^{(s+1)}}[\boldsymbol{w}_{new}] < \mathcal{J}_{\alpha^{(s+1)}}[\boldsymbol{w}^{(s)}]$
    - 3.1 Set $\boldsymbol{w}^{(s+1)} = \boldsymbol{w}_{new}$, $\eta = 0.5$, $s = s + 1$, and go to 4

    Else
    - 3.2 Set $\eta = 0.9$, and go to 4
  - 4. Check for convergence using the criterion (5.46)

    If not satisfied, then return to 1, else, exit to the next step to stop.
- Set $\boldsymbol{w}^* = \boldsymbol{w}_{new}$ and $\alpha^* = \alpha^{(s)}$.

Similarly, we use a hierarchy of L grids (with level L the finest and level 1 the coarsest one) with the multi-resolution technique in order to increase its performance and the whole procedure is summarised below:

Algorithm 6.5.2 (The refined RADPIR multi-resolution method)

---

1. Input $\overrightarrow{\varepsilon}_{lo}$ and $\overrightarrow{\varepsilon}_{hi}$.
2. Obtain the optimal regularisation parameter α (through cooling) and a good initial solution (through multi-resolution) $\boldsymbol{w}^{(0)}$:
   - $[\boldsymbol{w}^{(0)}, \alpha] \leftarrow RADPIR\_multiresolution(\boldsymbol{w}^{[L]}, \alpha^{[L]}, L, \overrightarrow{\varepsilon}_{lo})$
3. Solve the minimisation problem of (6.1) on the finest level $lev = L$ using the optimal value of α and the good initial solution $\boldsymbol{w}^{(0)}$:
   - $\boldsymbol{w}^{[lev]} \leftarrow FASNMG(\boldsymbol{w}^{(0)}, \alpha, \overrightarrow{\varepsilon}_{hi})$

---

where

$$\left[\boldsymbol{w}^{[lev]}, \alpha^{[lev]}\right] \leftarrow RADPIR\_multiresolution\left(\boldsymbol{w}^{[lev]}, \alpha^{[lev]}, lev, \overrightarrow{\varepsilon}\right)$$

- If $lev = 1$
  - $\boldsymbol{w}^{[lev]} = 0$
  - $\alpha^{[lev]} = C$ [$C > 0$ should be large enough e.g. $C = 100$]
  - $[\boldsymbol{w}^{[lev]}, \alpha^{[lev]}] \leftarrow cooling(\boldsymbol{w}^{[lev]}, \alpha^{[lev]}, \overrightarrow{\varepsilon})$
- Else
  - $\boldsymbol{w}^{[lev-1]} = (I_h^H w_1^{[lev]}, I_h^H w_2^{[lev]})^\top$
  - $\left[\boldsymbol{w}^{[lev-1]}, \alpha^{[lev-1]}\right] \leftarrow RADPIR\_multiresolution\left(\boldsymbol{w}^{[lev-1]}, \alpha^{[lev-1]}, lev-1, \overrightarrow{\varepsilon}\right)$
  - $\boldsymbol{w}^{[lev]} = (I_H^h w_1^{[lev-1]}, I_H^h w_2^{[lev-1]})^\top$
  - $\alpha^{[lev]} = 4\alpha^{[lev-1]}$ [Recall that $\alpha^{[lev]} = \overline{\alpha} n_{lev}^2$ and $n_{lel} = 2n_{lel-1}$]
  - $\boldsymbol{w}^{[lev]} \leftarrow FASNMG(\boldsymbol{w}^{[lev]}, \alpha^{[lev]}, \overrightarrow{\varepsilon})$
- Endif

6.6 Numerical experiments

In this section we demonstrate 3 sets of experimental results

(i) the abilities of $\mathcal{R}^{\text{MTV}}$ in solving the particular registration problems as represented by Example 1[1]-2[2] shown respectively in Figure 6.2 $(a)-(b)$ and Figure 6.4 $(a)-(b)$;

(ii) the overall performance of the RADPIR approach on two sets of medical data by processing Example 2 and 3 shown in Figure 6.4 $(a)-(b)$ and $(d)-(e)$;

(iii) a comparison between the RADPID approach and the semi-implicit time marching schemes as discussed in §6.3.3 on the set of clinical images in Example 2.

In all experiments, the bi-linear interpolation technique was used to compute $T(\boldsymbol{u})$, and $\nu_1 = 5$, $\nu_2 = 5$, $\omega = 1.85$, $GSiter = 5$ were employed in our FAS-NMG framework with a zero deformation field as initialisation at the finest level.

6.6.1 Comparison $\mathcal{R}^{\text{MTV}}$ with different regularisation techniques

In this experiment, our aim is to investigate capabilities of $\mathcal{R}^{\text{MTV}}$, $\mathcal{R}^{\beta\text{TV}}$ and $\mathcal{R}^{\text{diff}}$ (from §3.4), which belong to the same class of variational image registration models using 1^{st}-order derivatives in solving Example 1-2. To be a fair comparison, we used the same systematic formulation as explained in §6.3 – 6.4, in particular Algorithm 6.5.2, for solving the discretised Euler-Lagrange equations related to $\mathcal{R}^{\beta\text{TV}}$ and $\mathcal{R}^{\text{diff}}$.

As shown in Figure 6.2 $(c)-(e)$, on one hand, $\mathcal{R}^{\text{MTV}}$ and $\mathcal{R}^{\beta\text{TV}}$ produced visually pleasing registration results, while $\mathcal{R}^{\text{diff}}$ did not. The main reason is that the exact deformation field is given by a shift of the upper rectangular to the right and a shift of the lower rectangular to the left; c.f. Figure 6.3 $(a)-(b)$. Therefore, the exact deformation field is piecewise constant with substantial discontinuities at regions close to the interface between the upper and the lower rectangular. Consequently, $\mathcal{R}^{\text{diff}}$ must fail because it tries to smooth the deformation field as much as possible at those regions; see over smoothing results of the field as shown in Figure 6.3

[1] Adapted from [51] and [53]
[2] Source: http://www.math.mu-luebeck.de/safir/

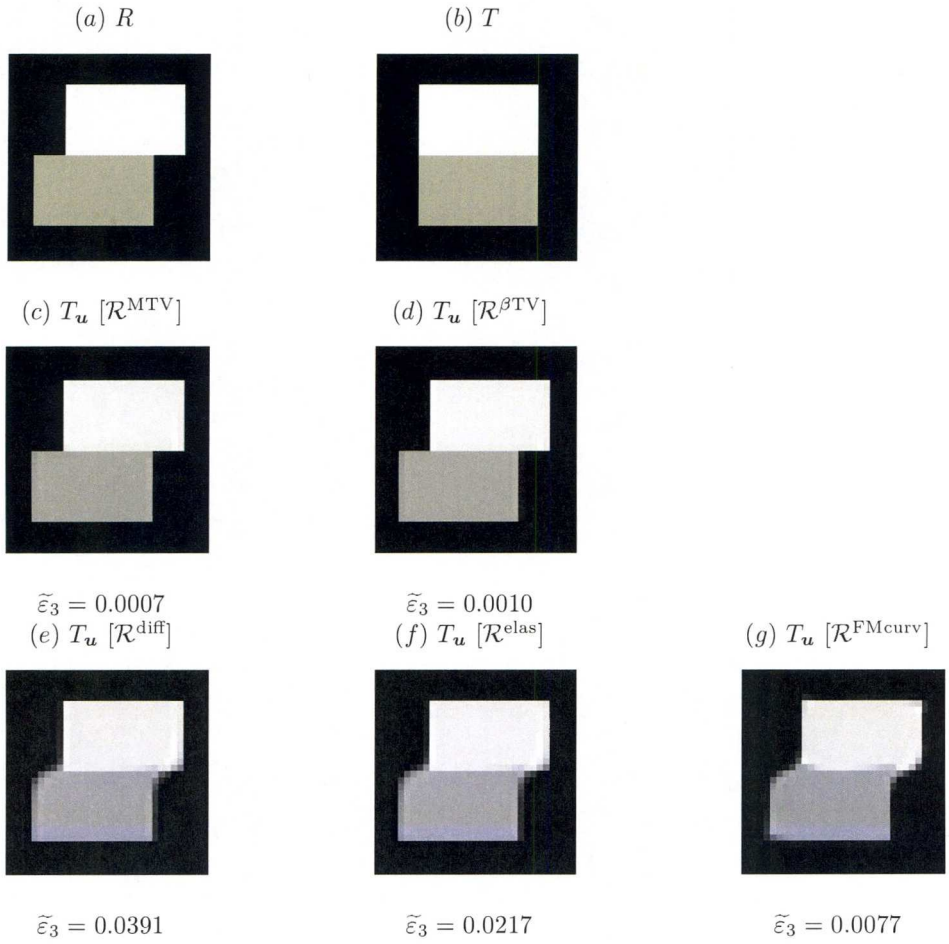


Figure 6.2: Registered images for two rectangular blocks shown in (a) R and (b) T of size 32×32 (Example 1): results by (c) $\mathcal{R}^{\mathrm{MTV}}$, (d) $\mathcal{R}^{\beta\mathrm{TV}}$ with $\beta = 0.0001$, (e) $\mathcal{R}^{\mathrm{diff}}$, (f) $\mathcal{R}^{\mathrm{elas}}$ with $(\mu, \lambda) = (1, 1)$, and (g) $\mathcal{R}^{\mathrm{FMcurv}}$. Recall that $\widetilde{\varepsilon}_3$ means the relative reduction of dissimilarity defined in Algorithm 6.4.1.

(c). On the other hand, as shown in Figures 6.5 $(a) - (c)$ $\mathcal{R}^{\mathrm{MTV}}$ and $\mathcal{R}^{\mathrm{diff}}$ gave slightly better registration results than those of $\mathcal{R}^{\beta\mathrm{TV}}$ in terms of $\widetilde{\varepsilon}_3$ (the relative reduction of dissimilarity), but the corresponding deformation fields shown in Figures 6.6 $(b) - (c)$ are more reasonable than that of $\mathcal{R}^{\beta\mathrm{TV}}$ depicted in Figures 6.6 (a). This is because the exact deformation field is globally smooth, almost the same shapes as determined by $\mathcal{R}^{\mathrm{MTV}}$ and $\mathcal{R}^{\mathrm{diff}}$, but the results of $\mathcal{R}^{\beta\mathrm{TV}}$ are almost piecewise constant in some parts of the upper regions. Both experiments confirm that $\mathcal{R}^{\mathrm{MTV}}$ is a half way between $\mathcal{R}^{\beta\mathrm{TV}}$ and $\mathcal{R}^{\mathrm{diff}}$. In other words, $\mathcal{R}^{\mathrm{MTV}}$ is compatible with $\mathcal{R}^{\beta\mathrm{TV}}$ for registration problems requiring to preserve discontinuities and it is compatible with $\mathcal{R}^{\mathrm{diff}}$ for those registration problems requiring to have global smoothness of the field. In case of $\mathcal{R}^{\mathrm{elas}}$ and $\mathcal{R}^{\mathrm{FMcurv}}$, we found by applying different numerical techniques given in [104] that registration results are similar to those of $\mathcal{R}^{\mathrm{diff}}$ as shown in Figures 6.2 $(f) - (g)$, 6.3 $(d) - (e)$, 6.5 $(d) - (e)$, and 6.6 $(d) - (e)$.

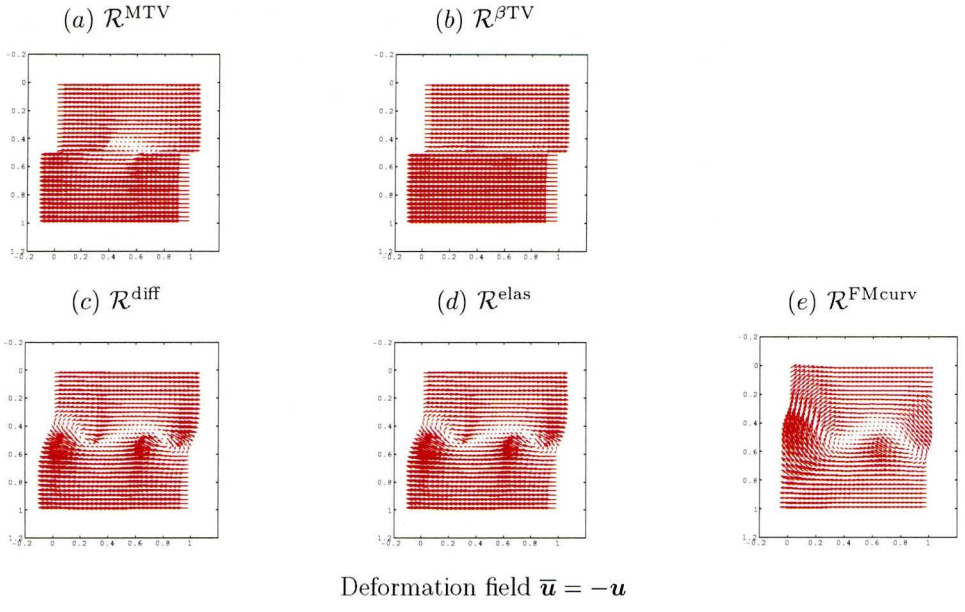


Figure 6.3: Deformation fields for the registration problem shown in Figure 6.2 (a)-(b) (Example 1): results by (a) $\mathcal{R}^{\mathrm{MTV}}$, (b) $\mathcal{R}^{\beta\mathrm{TV}}$ with $\beta = 0.0001$, (c) $\mathcal{R}^{\mathrm{diff}}$, (d) $\mathcal{R}^{\mathrm{elas}}$ with $(\mu, \lambda) = (1, 1)$, and (e) $\mathcal{R}^{\mathrm{FMcurv}}$.

6.6.2 h−independent convergent tests for Algorithms 6.4.1, 6.5.1, and 6.5.2

One of the key properties of multigrid techniques is that their convergence does not depend on the number of grid points. Thus, in the second test we designed our experiments on clinical images by processing Example 2 and 3 as shown in Figure 6.4 $(a) - (b)$ and $(d) - (e)$ with Algorithms 6.4.1, 6.5.1, and 6.5.2. The number of multigrid steps (V-cycles) used to drop the relative residual below 10^{-8}, the relative reduction of dissimilarity, and the run times (in seconds) are given in Table 6.1 with different sizes of grid points. The results show that all registration algorithms not only converge within a few multigrid steps as expected from a multigrid technique, but they are also accurate because the dissimilarities between the reference and registered images have been reduced more than 88% for Example 1 and 94% for Example 2. For overall performance the experimental results suggest that Algorithm 6.5.2 would be preferred for practical applications because the multi-resolution idea used in cooling α has proved to be very useful for initialisation. It results in speeding up overall run times of Algorithm 6.5.1 around 3 times.

6.6.3 Comparison Algorithm 6.4.1 with two time-marching methods

The main aim of this experiment is to show that the parabolic approach is quite slow in achieving convergence. We took Example 2 to illustrate this point. Table 6.2 summarises the results for the standard semi-implicit and AOS time marching schemes with different numbers of grid

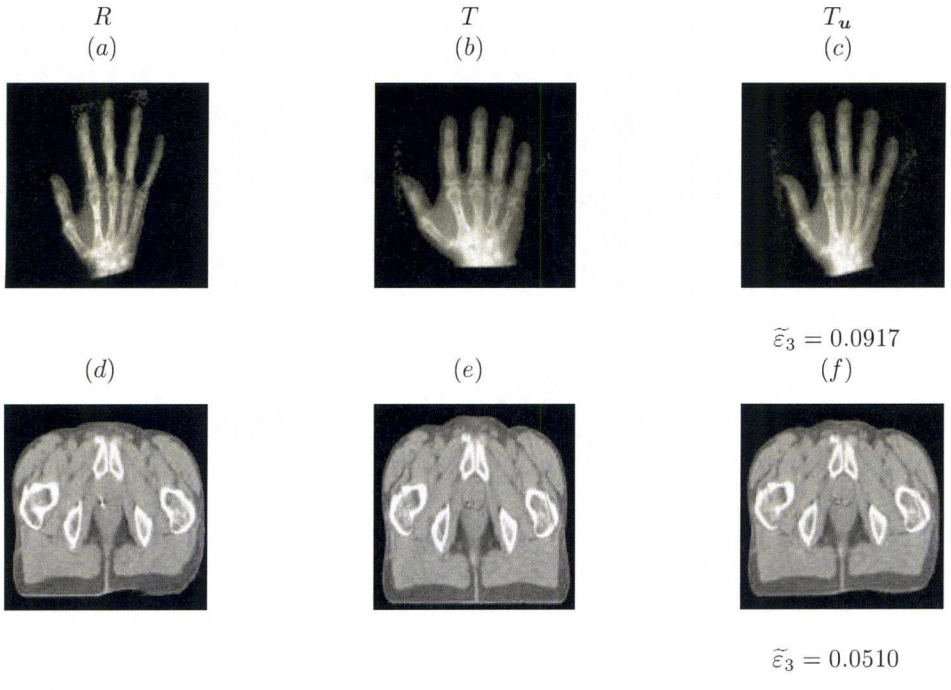


Figure 6.4: Registration results for X-ray and MRI images (Examples 2 $(a) - (b)$and 3 $(d) - (e)$) using the RADPIR approach with Algorithms 6.4.1, 6.5.1, and 6.5.2. Left column: reference R, center column: template T, right column: the deformed template image $T(\boldsymbol{u})$ obtained from RADPIR.

points. To be a fair comparison between them, we used those results determined by Algorithm 6.4.1 as shown in Table 6.1. That is, we started all methods with the same $\alpha = 0.0909$ and the same initial guess, $\boldsymbol{u}^{(0)} = 0$. Here, the time-step τ is required to be sufficiently small for each size of the problem. We used $\tau = 10^{-2}$ for $h = 1/128 - 1/1024$. As expected from the experiments, all methods are accurate in registering the given images because the dissimilarities between the reference and registered images have been reduced more than 88%. However both time-marching methods fail to drop the relative residual to 10^{-8} in a few time steps (even large values of τ are used) and the run times used by Algorithm 6.4.1 are significantly faster in delivering the same level of the relative dissimilarity.

6.7 Conclusion

In this chapter, we proposed first a novel discontinuity-preserving image registration model, which can be viewed as a hybrid model between the diffusion and TV models, for solving both smooth and non-smooth registration problems. Second, we introduced a fully automatic, fast, and accurate approach based on the FAS-NMG strategy and the automatic procedure in selecting the optimal α in order to solve the resulting Euler-Lagrange system. Numerical tests from the previous section confirmed that the proposed model is more flexible than the

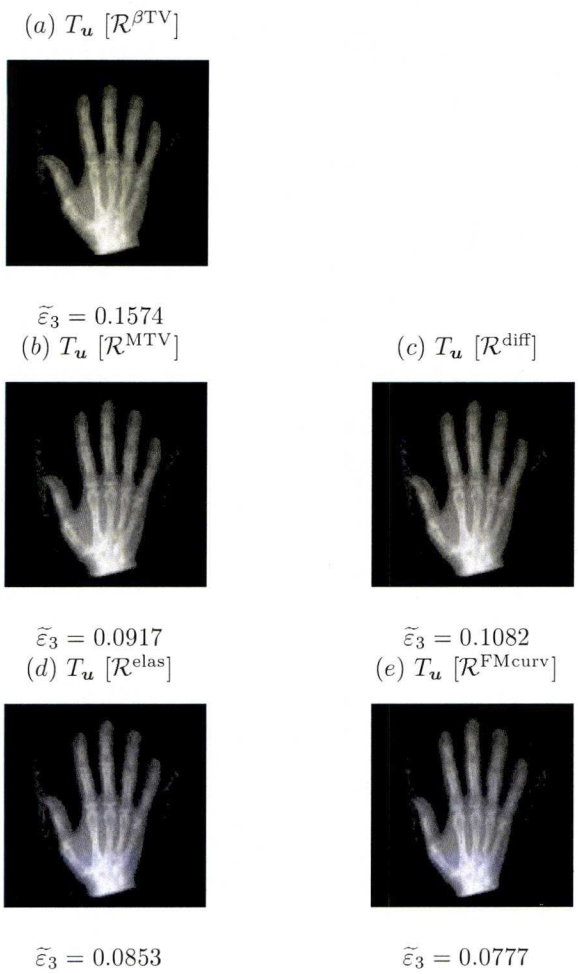


Figure 6.5: Registration results for the problem of size 128×128 shown in Figure 6.4 (a)-(b) (Example 2): results by (a) $\mathcal{R}^{\beta\mathrm{TV}}$ with $\beta = 0.001$, (b) $\mathcal{R}^{\mathrm{MTV}}$, (c) $\mathcal{R}^{\mathrm{diff}}$, (d) $\mathcal{R}^{\mathrm{elas}}$ with $(\mu, \lambda) = (1, 1)$, and (e) $\mathcal{R}^{\mathrm{FMcurv}}$.

diffusion and TV models. Moreover, they also showed that the FAS-NMG technique based on the proposed FP type smoother is $h-$independent convergence and much faster than those of standard unilevel methods such as semi-implicit time marching and AOS schemes in convergence and delivering the same numerical results.

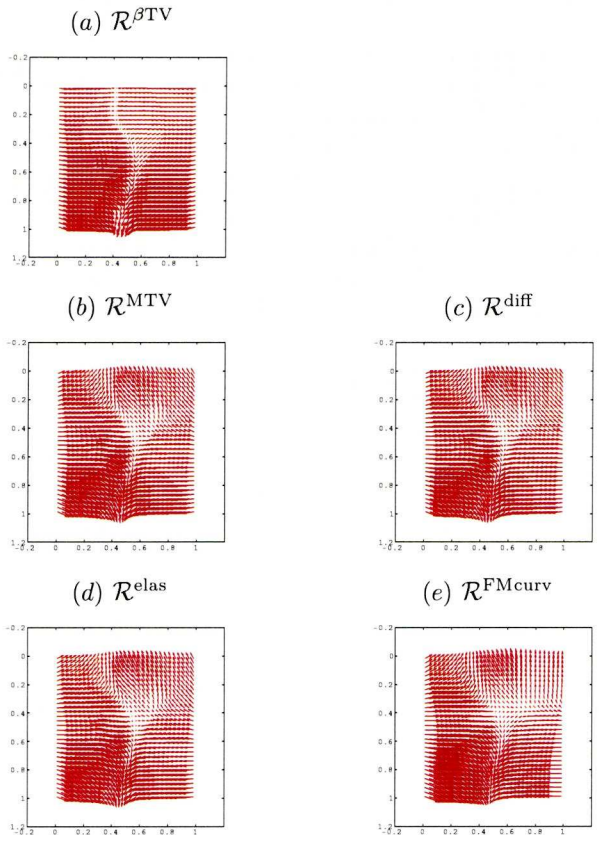


Figure 6.6: Deformation fields for the registration problem shown in Figure 6.4 (a)-(b) (Example 2): results by (a) $\mathcal{R}^{\beta\mathrm{TV}}$ with $\beta = 0.001$, (b) $\mathcal{R}^{\mathrm{MTV}}$, (c) $\mathcal{R}^{\mathrm{diff}}$, (d) $\mathcal{R}^{\mathrm{elas}}$ with $(\mu, \lambda) = (1, 1)$, and (e) $\mathcal{R}^{\mathrm{FMcurv}}$.

	Algorithm 6.4.1 M/R/D/C	**Algorithm 6.5.1** M/R/D/IC/C	**Algorithm 6.5.2** M/R/D/IC/C
Example 2 :	$\alpha = 0.0909$		
$h = 1/128$	$8/3.1 \times 10^{-9}/0.1012/26.2$	$4/6.6 \times 10^{-9}/0.0917/95.4/110.1$	$6/7.1 \times 10^{-9}/0.0917/21.6/41.0$
$h = 1/256$	$8/1.8 \times 10^{-9}/0.1098/134.4$	$5/3.6 \times 10^{-9}/0.1098/298.0/365.1$	$6/3.1 \times 10^{-9}/0.1098/29.4/109.5$
$h = 1/512$	$8/9.5 \times 10^{-9}/0.1150/453.3$	$5/1.4 \times 10^{-9}/0.1124/1402.1/1707.5$	$6/4.4 \times 10^{-9}/0.1124/57.9/396.1$
$h = 1/1024$	$8/3.6 \times 10^{-9}/0.1168/1864.4$	$5/2.1 \times 10^{-9}/0.1168/5137.5/6289.7$	$5/6.5 \times 10^{-9}/0.1168/171.9/1332.1$
Example 3 :	$\alpha = 0.1111$		
$h = 1/128$	$7/8.1 \times 10^{-9}/0.0528/22.9$	$4/1.8 \times 10^{-9}/0.0510/69.9/84.6$	$6/2.8 \times 10^{-9}/0.0510/18.5/38.0$
$h = 1/256$	$9/3.8 \times 10^{-9}/0.0595/121.7$	$5/1.9 \times 10^{-9}/0.0510/312.2/386.4$	$7/3.1 \times 10^{-9}/0.0510/27.6/121.9$
$h = 1/512$	$10/3.8 \times 10^{-9}/0.0592/566.2$	$5/7.8 \times 10^{-9}/0.0544/1274.6/1578.0$	$7/7.1 \times 10^{-9}/0.0544/58.9/455.6$
$h = 1/1024$	$10/5.5 \times 10^{-9}/0.0591/2447.9$	$6/2.9 \times 10^{-9}/0.0591/4678.5/6069.2$	$7/2.2 \times 10^{-9}/0.0592/189.0/1905.3$

Table 6.1: Registration results of Algorithms 6.4.1, 6.5.1, and 6.5.2 for processing Examples 2 and 3 shown respectively in Figure 3 $(a) - (b)$and $(d) - (e)$. The letters 'M', 'R', 'D', 'C', and 'IC' mean the number of multigrid steps, the relative reduction of residual, the relative reduction of dissimilarity, the total run times (in seconds), and the initial run times (in seconds) for determining the optimal α and initial guess $\boldsymbol{u}^{(0)}$, respectively.

	SITM (6.15) M/R/D/C	**AOS (6.16)** M/R/D/C
$h = 1/128$	21973/ $*$ /0.1012/5232.4 (1.45 hours)	23946/ $*$ /0.1012/3074.1 (0.85 hours)
$h = 1/256$	19808/ $*$ /0.1098/25513.9 (7.08 hours)	21197/ $*$ /0.1098/15587.0 (4.32 hours)
$h = 1/512$	$*/*/*/*$ ($>$ 10 hours)	$*/*/*/*$ ($>$ 10 hours)
$h = 1/1024$	$*/*/*/*$ ($>$ 10 hours)	$*/*/*/*$ ($>$ 10 hours)

Table 6.2: Registration results of the SITM and AOS methods, represented in (6.15) and (6.16) for Example 2 shown in Figure 6.4 $(a) - (b)$. $*$ indicates either computation stopped after about 10 hours or failure in dropping the relative residual to 10^{-8}.

Chapter 7

A Fourth Order Variational Image Registration Model and Its Fast Multigrid Algorithm

Several PDE-based variational methods can be used for deformable image registration, mainly differing in how regularisation to constrain deformation fields is imposed [104]. As mentioned in the previous chapter, on one hand for smooth registration problems, models of elastic-, diffusion-, and curvature-based image registration are known to generate globally smooth and satisfactory deformation fields. On the other hand for non-smooth registration problems, models based on the total variation (TV) regularisation are better for preserving discontinuities of the deformation fields.

In this chapter we propose and study a promising model that is based on a novel curvature type regulariser and appears to deliver excellent results for both registration problems. A related work due to Fischer and Modersitzki [47] and then refined by Henn and Witsch [78] used an approximation of the mean curvature and obtained improved results over previous models. However, this chapter investigates the full curvature model and finds that the new curvature model is more robust than approximated curvature models and leads to further improvement.

Associated with the new model is the apparent difficulty in developing a fast solution as the system of two coupled PDEs is highly nonlinear and of fourth order so standard application of multigrid methods does not work. To end this, we first propose several fixed-point type smoothers and use both local Fourier analysis and numerical experiments to select the most effective smoother which turns out to be a primal-dual based method. Finally we use the recommended smoother with a FAS-NMG algorithm for the new model. Numerical tests using both synthetic and realistic images not only confirm that the proposed curvature model is more robust in registration quality for a wide range of applications than previous work [104, 78], but also that the proposed numerical algorithm is fast and accurate in delivering visually-pleasing registration results.

7.1 Introduction

Let R and T denote a reference and a template image, respectively. Here the given images R and T are modelled as the continuous functions mapping from an image domain $\Omega = [0,1]^2 \subset \mathbb{R}^2$ into $V = [0,1] \subset \mathbb{R}_0^+$. As already pointed out in §3.2, the registration problem can be posed as the following minimisation problem:

$$\min_{\boldsymbol{u}} \{\mathcal{J}_\alpha(\boldsymbol{u}) = \mathcal{D}^{\mathrm{SSD}}(R, T_{\boldsymbol{u}}) + \alpha \mathcal{R}(\boldsymbol{u})\} \tag{7.1}$$

where the image intensities of the given images R and T are assumed to be comparable and $\alpha > 0$ is the regularisation parameter. Recall that

$$\mathcal{D}^{\mathrm{SSD}}(\boldsymbol{u}) = \frac{1}{2}\int_\Omega (T(\mathbf{x} + \boldsymbol{u}(\mathbf{x})) - R(\mathbf{x}))^2 \, d\mathbf{x}, \tag{7.2}$$

and

$$\boldsymbol{f}(\boldsymbol{u}) = (f_1(\boldsymbol{u}), f_2(\boldsymbol{u}))^\top = ((T_{\boldsymbol{u}} - R)\,\partial_{u_1} T_{\boldsymbol{u}}, (T_{\boldsymbol{u}} - R)\,\partial_{u_2} T_{\boldsymbol{u}})^\top \tag{7.3}$$

is related to the first variation of $\mathcal{D}^{\mathrm{SSD}}$.

Below we review the specific choice of $\mathcal{R}$ and the subsequent system in five commonly used PDE models.

(1) *Elastic image registration* [8, 15, 104]: Choosing $\mathcal{R}$ in (7.1) by

$$\mathcal{R}^{\mathrm{elas}}(\boldsymbol{u}) = \int_\Omega ((\mu/4) \sum_{l,m=1}^{2} (\partial_{x_l} u_m + \partial_{x_m} u_l)^2 + (\lambda/2)(\nabla \cdot \boldsymbol{u})^2) d\mathbf{x}, \tag{7.4}$$

leads to the Euler-Lagrange system of two second-order nonlinear PDEs:

$$\begin{cases} f_1(\boldsymbol{u}) - \alpha((\lambda+2\mu)\partial_{x_1x_1}u_1 + \mu\partial_{x_2x_2}u_1 + (\lambda+\mu)\partial_{x_1x_2}u_2) = 0 \\ f_2(\boldsymbol{u}) - \alpha((\lambda+\mu)\partial_{x_1x_2}u_1 + \mu\partial_{x_1x_1}u_2 + (\lambda+2\mu)\partial_{x_2x_2}u_2) = 0, \end{cases} \text{(elastic model)} \tag{7.5}$$

subject to $\langle \mu(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^\top) + \lambda \mathrm{diag}(\nabla \cdot \boldsymbol{u}), \boldsymbol{n} \rangle_{\mathbb{R}^2} = 0$ on $\partial\Omega$.

(2) *Diffusion image registration* [33, 46, 89, 91, 104, 131]: Choosing $\mathcal{R}$ in (7.1) by

$$\mathcal{R}^{\mathrm{diff}}(\boldsymbol{u}) = \frac{1}{2}\sum_{l=1}^{2}\int_\Omega |\nabla u_l|^2 \, d\mathbf{x}, \tag{7.6}$$

leads to the Euler-Lagrange system of two second-order nonlinear PDEs:

$$\begin{cases} f_1(\boldsymbol{u}) - \alpha\Delta u_1 = 0 \\ f_2(\boldsymbol{u}) - \alpha\Delta u_2 = 0 \end{cases} \text{(diffusion model)} \tag{7.7}$$

subject to $\langle \nabla u_l, \boldsymbol{n} \rangle_{\mathbb{R}^2} = 0$ on $\partial\Omega$.

(3) *Fischer–Modersitzki's curvature image registration* [47, 48, 49, 89, 91, 104]: Choosing $\mathcal{R}$ in (7.1) by

$$\mathcal{R}^{\mathrm{FMcurv}}(\boldsymbol{u}) = \frac{1}{2}\sum_{l=1}^{2}\int_\Omega (\Delta u_l)^2 d\mathbf{x}, \tag{7.8}$$

leads to the Euler-Lagrange system of two fourth-order nonlinear PDEs:

$$\begin{cases} f_1(\boldsymbol{u}) + \alpha\Delta^2 u_1 = 0 \\ f_2(\boldsymbol{u}) + \alpha\Delta^2 u_2 = 0 \end{cases} \text{(\textit{Fischer–Modersitzki's} curvature model)} \tag{7.9}$$

subject to the special boundary conditions $\nabla u_l = 0$, $\nabla\Delta u_l = 0$ on $\partial\Omega$, for $l = 1, 2$.

(4) *Henn–Witsch's curvature image registration* [79, 78, 73, 75, 74]. Choosing $\mathcal{R}$ in (7.1) by

$$\mathcal{R}^{\mathrm{HWcurv}}(\boldsymbol{u})=\frac{1}{2}\sum_{l=1}^{2}\int_{\Omega}(\Delta u_l)^2-2(u_{l_{x_1x_1}}u_{l_{x_2x_2}}-u_{l_{x_1x_2}}^2)d\mathbf{x}, \tag{7.10}$$

leads to the Euler-Lagrange system of two modified fourth-order nonlinear PDEs:

$$\begin{cases} f_1(\boldsymbol{u})+\alpha\Delta^2 u_1=0 \\ f_2(\boldsymbol{u})+\alpha\Delta^2 u_2=0 \end{cases} \quad (\textit{Henn–Witsch's} \text{ curvature model}) \tag{7.11}$$

subject to $\mathcal{B}_l(u_l)=0$ on $\partial\Omega$ with

$$\mathcal{B}_1(u_l)=-\frac{\partial}{\partial\boldsymbol{n}}\Delta u_l-\frac{\partial}{\partial\boldsymbol{s}}\left[\frac{\partial^2 u_l}{\partial x_1\partial x_2}(n_1^2-n_2^2)+\left(\frac{\partial^2 u_l}{\partial^2 x_2}-\frac{\partial^2 u_l}{\partial^2 x_1}\right)n_{x_1}n_{x_2}\right],$$

and

$$B_2(u_l)=\frac{\partial^2 u_l}{\partial\boldsymbol{n}^2}$$

where $\boldsymbol{s}$ denotes the unit tangential vector (orthogonal to $\boldsymbol{n}$).

(5) *Total variation (TV) image registration* [51, 53, 142]: Choosing $\mathcal{R}$ in (7.1) by

$$\mathcal{R}^{\beta\mathrm{TV}}(\boldsymbol{u})=\sum_{l=1}^{2}\int_{\Omega}|\nabla u_l|_\beta\, d\mathbf{x}, \tag{7.12}$$

leads to the Euler-Lagrange system of two second-order nonlinear PDEs:

$$\begin{cases} f_1(\boldsymbol{u})-\alpha\nabla\cdot\left(\frac{\nabla u_1}{|\nabla u_1|_\beta}\right)=0 \\ f_2(\boldsymbol{u})-\alpha\nabla\cdot\left(\frac{\nabla u_2}{|\nabla u_2|_\beta}\right)=0 \end{cases} \quad (\text{TV model}) \tag{7.13}$$

subject to $\langle\nabla u_l,\boldsymbol{n}\rangle_{\mathbb{R}^2}=0$ on $\partial\Omega$.

As pointed out several times in the previous chapters, the first four models are quite different from the fifth one. Firstly, $\mathcal{R}^{\mathrm{elas}}$, $\mathcal{R}^{\mathrm{diff}}$, $\mathcal{R}^{\mathrm{FMcurv}}$, and $\mathcal{R}^{\mathrm{HWcurv}}$ produce globally smooth deformation fields, although the latter two models are better than the former two. While they are useful for several applications, they become poor if discontinuities or steep gradients in the deformation fields are expected (e.g. resulting from matching several moved objects or partially occluded objects). See Figures 7.1-7.2 for a particular registration problem where these regularisation techniques yield oversmooth deformation fields.

Secondly, $\mathcal{R}^{\beta\mathrm{TV}}$ helps to preserve discontinuities of the deformation field in clear contrast to the first four models; see Figures 7.3-7.4 for example, in particular the piecewise smoothness shown in Figure 7.4 (*c*) at the top region. However, $\mathcal{R}^{\beta\mathrm{TV}}$ may not be suitable for smooth registration problems, which are modelled better with the first four methods.

In addition to these five models, the optical flow models [2, 3, 18] are also widely used which works the best if features have minor changes from R to T, e.g. in matching sequential frames in a video.

As is well-known, efficient solution of the coupled nonlinear PDEs resulting from a variational registration model is an important task. For registration purposes, various numerical techniques based on unilevel and multilevel methods have been proposed and tested as briefly reviewed in

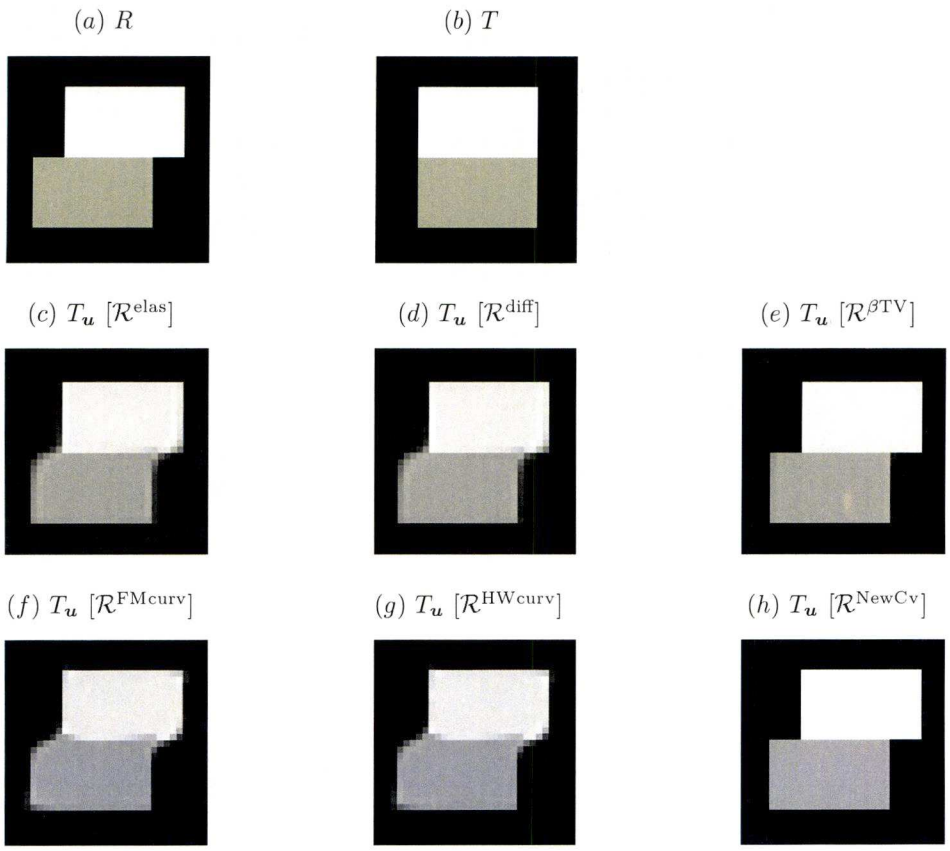


Figure 7.1: Registered images for two rectangular blocks shown in (*a*) R and (*b*) T of size 32×32 (Example 1: results by (*c*) $\mathcal{R}^{\text{elas}}$ with $(\mu, \lambda) = (1, 1)$, (*d*) $\mathcal{R}^{\text{diff}}$, (*e*) $\mathcal{R}^{\beta\text{TV}}$ with $\beta = 0.01$, (*f*) $\mathcal{R}^{\text{FMcurv}}$, (*g*) $\mathcal{R}^{\text{HWcurv}}$, (*h*) $\mathcal{R}^{\text{NewCv}}$ with $\beta = 0.01$. A non-smooth deformation example to show that our registration model $\mathcal{R}^{\text{NewCv}}$ gives the satisfactory registration results as good as those from $\mathcal{R}^{\beta\text{TV}}$, which is known to be suitable. Here the regularisation parameter α was well-selected for all registration models.

§3.6. However, the new fourth-order model to be proposed here cannot be solved by existing methods. The new algorithms will be presented shortly.

The rest of the chapter is organized as follows. §7.2 first presents a new PDE-based image registration model based on a novel curvature regulariser suitable for both smooth and non-smooth deformation problems and then discusses unilevel iterative numerical methods for it in §7.3. §7.4 presents a fast multigrid approach after first analysing some iterative solvers as potential smoothers. Experimental results from real images illustrating the improved results from the new model and the efficiency from FAS-NMG are shown in §7.5 before conclusions in §7.6.

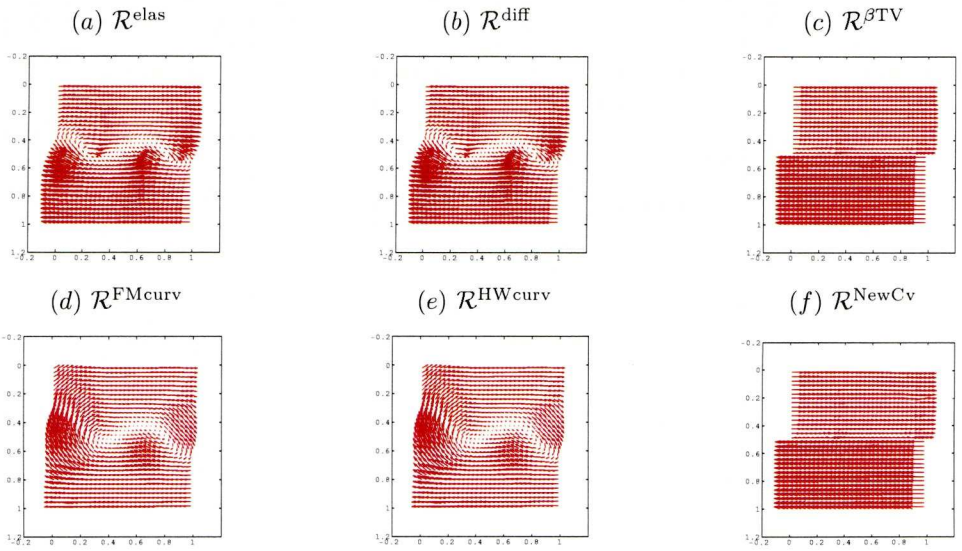


Figure 7.2: Deformation fields for the non-smooth registration problem shown in Figure 7.1 (a)-(b) (Example 1): results by (a) $\mathcal{R}^{\text{elas}}$ with $(\mu, \lambda) = (1, 1)$, (b) $\mathcal{R}^{\text{diff}}$, (c) $\mathcal{R}^{\beta\text{TV}}$ with $\beta = 0.01$, (d) $\mathcal{R}^{\text{FMcurv}}$, (e) $\mathcal{R}^{\text{HWcurv}}$, and (f) $\mathcal{R}^{\text{NewCv}}$ with $\beta = 0.01$. The exact deformation field is given by a shift of the upper rectangular to the right and a shift of the lower rectangular to the left; c.f. Figure 7.1 $(a) - (b)$.

7.2 A new PDE-based image registration model

Motivated by the attractive properties of the Fischer–Modersitzki's curvature registration model (7.8) improving on previous second order models (7.5) and (7.7), we consider an alternative formulation that uses the full curvature information without approximations and hope to achieve further improvements in terms of registration quality. It turns out that a model that minimises the curvatures along level lines is the right model to study while a model uses the (mean) curvature

$$\kappa_M(u_l) = \nabla \cdot \frac{\nabla u_l}{\sqrt{1+|\nabla u_l|^2}} = \frac{(1+u_{l_{x_1}}^2)u_{l_{x_1x_1}} - 2u_{l_{x_1}}u_{l_{x_2}}u_{l_{x_1x_2}} + (1+u_{l_{x_2}}^2)u_{l_{x_2x_2}}}{(1+u_{l_{x_1}}^2+u_{l_{x_2}}^2)^{3/2}}$$

cannot achieve such an aim.

Instead of using $\kappa_M(u_l)$, we consider the curvature of the level lines to allow displacement discontinuities

$$\kappa(u_l) = \nabla \cdot \frac{\nabla u_l}{|\nabla u_l|_\beta} = \frac{(\beta+u_{l_{x_1}}^2)u_{l_{x_1x_1}} - 2u_{l_{x_1}}u_{l_{x_2}}u_{l_{x_1x_2}} + (\beta+u_{l_{x_2}}^2)u_{l_{x_2x_2}}}{(\beta+u_{l_{x_1}}^2+u_{l_{x_2}}^2)^{3/2}},$$

and propose the following regulariser

$$\mathcal{R}^{\text{NewCv}}(\boldsymbol{u}) = \sum_{l=1}^{2} \int_\Omega \Phi(\kappa(u_l))d\mathbf{x}.$$

Due to the sum rule, the following theorem can be used to compute the first variation of $\mathcal{R}^{\text{NewCv}}$.

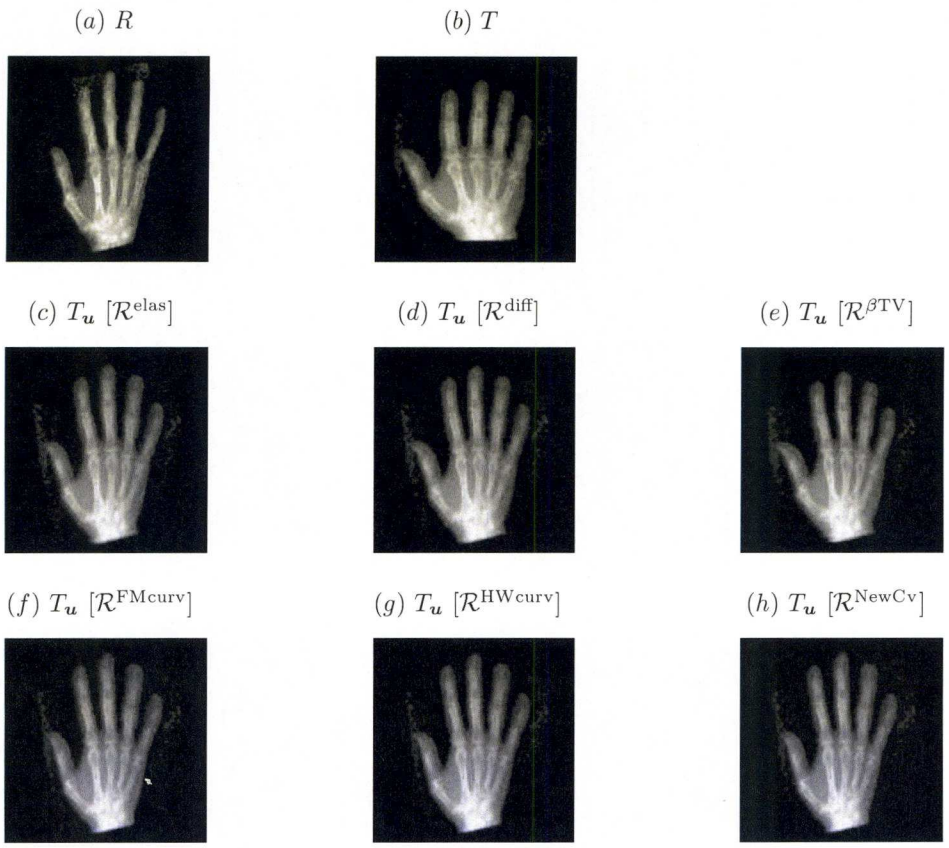


Figure 7.3: Registered images for X-ray images shown in (a) R and (b) T of size 128×128 (Example 2): results by (c) $\mathcal{R}^{\text{elas}}$ with $(\mu, \lambda) = (1, 1)$, (d) $\mathcal{R}^{\text{diff}}$, (e) $\mathcal{R}^{\beta\text{TV}}$ with $\beta = 0.01$, (f) $\mathcal{R}^{\text{FMcurv}}$, (g) $\mathcal{R}^{\text{HWcurv}}$, (h) $\mathcal{R}^{\text{NewCv}}$. A smooth deformation example to show that our registration model $\mathcal{R}^{\text{NewCv}}$ gives the satisfactory registration results as good as those from $\mathcal{R}^{\text{FMcurv}}$ and $\mathcal{R}^{\text{HWcurv}}$, which are known to be suitable. Here the regularisation parameter α was well-selected for all registration models.

Theorem 7.2.1 *Let Φ be a given function and let*

$$\overline{\mathcal{R}}^{NewCv}(u_l) = \int_\Omega \Phi(\kappa(u_l))d\mathbf{x}.$$

Then the first variation of $\overline{\mathcal{R}}^{NewCv}(u_l)$ is given by

$$\begin{aligned}\delta\overline{\mathcal{R}}^{NewCv}(u_l;\eta_l) = &\int_\Omega (\nabla\cdot(\frac{1}{|\nabla u_l|}\nabla\Phi'(\kappa(u_l)) - \frac{\nabla u_l\cdot\nabla\Phi'(\kappa(u_l))}{(|\nabla u_l|)^3}\nabla u_l))\eta_l d\mathbf{x}\\ &+\int_{\partial\Omega}\Phi'(\kappa(u_l))\left\langle\frac{1}{|\nabla u_l|}(\mathbf{I}-\mathbf{P})\nabla\eta_l,\boldsymbol{n}\right\rangle_{\mathbb{R}^2}d\boldsymbol{s}\\ &-\int_{\partial\Omega}\left\langle\frac{1}{|\nabla u_l|}(\mathbf{I}-\mathbf{P})\nabla(\Phi'(\kappa(u_l)),\boldsymbol{n}\right\rangle_{\mathbb{R}^2}\eta_l d\boldsymbol{s},\end{aligned} \tag{7.14}$$

where $\mathbf{I}$ is the identity transform and $\mathbf{P} = \nabla u_l\frac{\nabla u_l}{|\nabla u_l|^2} = \overrightarrow{\boldsymbol{n}}\otimes\overrightarrow{\boldsymbol{n}}$ is the orthogonal projection onto the normal direction.

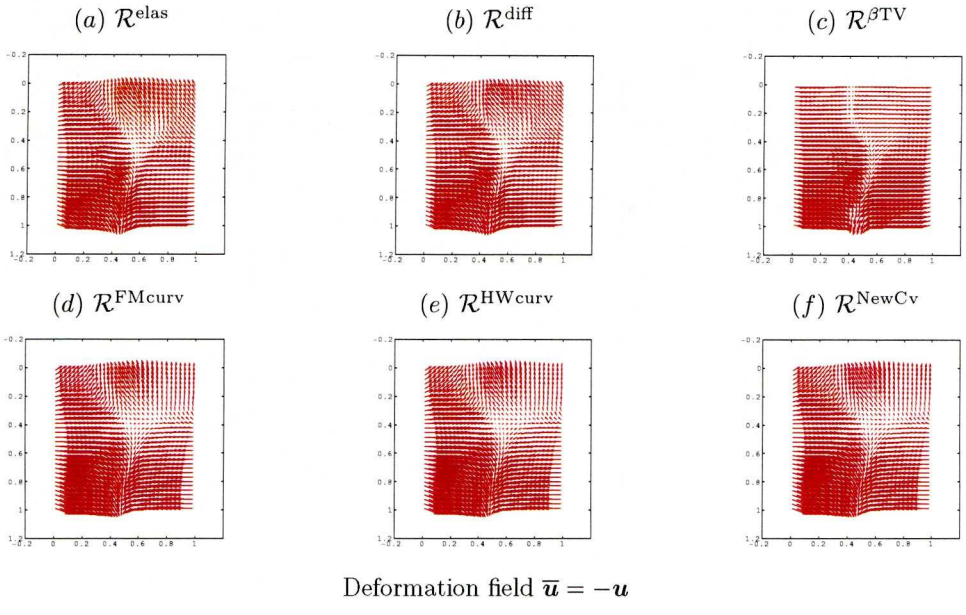


Figure 7.4: Deformation fields for the smooth registration problem shown in Figure 7.3 (*a*)-(*b*) (Example 2): results by (*a*) $\mathcal{R}^{\text{elas}}$ with $(\mu, \lambda) = (1, 1)$, (*b*) $\mathcal{R}^{\text{diff}}$, (*c*) $\mathcal{R}^{\beta\text{TV}}$ with $\beta = 0.01$, (*d*) $\mathcal{R}^{\text{FMcurv}}$, (*e*) $\mathcal{R}^{\text{HWcurv}}$, and (*f*) $\mathcal{R}^{\text{NewCv}}$. (*c*) shows the piecewise constant smoothness at the top region by $\mathcal{R}^{\beta\text{TV}}$.

Proof.

$$
\begin{aligned}
\delta\overline{\mathcal{R}}^{\text{NewCv}}(u_l;\eta_l) &= \frac{d}{d\epsilon}\overline{\mathcal{R}}^{\text{NewCv}}(u_l+\epsilon\eta_l)\Big|_{\epsilon=0} \\
&= \frac{d}{d\epsilon}\int_\Omega \Phi(\kappa(u_l+\epsilon\eta_l))d\mathbf{x}\Big|_{\epsilon=0} \\
&= \int_\Omega \frac{d}{d\epsilon}\Phi(\kappa(u_l+\epsilon\eta_l))\Big|_{\epsilon=0} d\mathbf{x} \\
&= \int_\Omega \Phi'(\kappa(u_l+\epsilon\eta_l))\frac{d}{d\epsilon}\kappa(u_l+\epsilon\eta_l)\Big|_{\epsilon=0} d\mathbf{x} \\
&= \int_\Omega \Phi'(\kappa(u_l))\frac{d}{d\epsilon}(\nabla\cdot\frac{\nabla(u_l+\epsilon\eta_l)}{|\nabla(u_l+\epsilon\eta_l)|})\Big|_{\epsilon=0} d\mathbf{x} \\
&= \int_\Omega \Phi'(\kappa(u_l))(\nabla\cdot\frac{d}{d\epsilon}\frac{\nabla(u_l+\epsilon\eta_l)}{|\nabla(u_l+\epsilon\eta_l)|}\Big|_{\epsilon=0})d\mathbf{x}
\end{aligned}
$$

Figure 7.5: Surface plots of u_1 for the non-smooth registration problem shown in Figure 7.1 (*a*)-(*b*) (Example 1): results by $\mathcal{R}^{\text{NewCv}}$ with (*a*) $\beta = 1$ and (*b*) $\beta = 0.01$. (*a*) and (*b*) show smoothing effects on the surface of u_1 at two different values of β.

or

$$\begin{aligned}
\delta\overline{\mathcal{R}}^{\text{NewCv}}(u_l;\eta_l) &= \int_\Omega \Phi'(\kappa(u_l))\left(\nabla\cdot\left(\frac{1}{|\nabla(u_l+\epsilon\eta_l)|}\frac{d}{d\epsilon}\nabla(u_l+\epsilon\eta_l)\Big|_{\epsilon=0}\right.\right.\\
&\qquad\left.\left.+\nabla(u_l+\epsilon\eta_l)\frac{d}{d\epsilon}\frac{1}{|\nabla(u_l+\epsilon\eta_l)|}\Big|_{\epsilon=0}\right)\right)d\mathbf{x}\\
&= \int_\Omega \Phi'(\kappa(u_l))\left(\nabla\cdot\left(\frac{\nabla\eta_l}{|\nabla u_l|}+\nabla u_l\sum_{m=1}^{2}\frac{\partial[(u_{l_{x_1}}+\epsilon\eta_{lx_1})^2+(u_{l_{x_2}}+\epsilon\eta_{lx_2})^2]^{-1/2}}{\partial(u_{l_{x_m}}+\epsilon\eta_{lx_m})}\times\right.\right.\\
&\qquad\left.\left.\frac{\partial(u_{l_{x_m}}+\epsilon\eta_{lx_m})^2}{\partial(u_{l_{x_m}}+\epsilon\eta_{lx_m})}\frac{\partial(u_{l_{x_m}}+\epsilon\eta_{lx_m})}{\partial\epsilon}\Big|_{\epsilon=0}\right)\right)d\mathbf{x}\\
&= \int_\Omega \Phi'(\kappa(u_l))(\nabla\cdot(\frac{\nabla\eta_l}{|\nabla u_l|}-\nabla u_l\sum_{m=1}^{2}\frac{u_{l_{x_m}}\eta_{l_{x_m}}}{|\nabla u_l|^3}))d\mathbf{x}\\
&= \int_\Omega \Phi'(\kappa(u_l))(\nabla\cdot(\frac{\nabla\eta_l}{|\nabla u_l|}-(\nabla u_l\frac{\nabla u_l}{|\nabla u_l|^3})\nabla\eta_l))d\mathbf{x}.
\end{aligned}$$

After applying the divergence theorem, we get

$$\begin{aligned}
\delta\overline{\mathcal{R}}^{\text{NewCv}}(u_l;\eta_l) &= \int_{\partial\Omega}\Phi'(\kappa(u_l))\left\langle\frac{\nabla\eta_l}{|\nabla u_l|}-(\nabla u_l\frac{\nabla u_l}{|\nabla u_l|^3})\nabla\eta_l,\boldsymbol{n}\right\rangle_{\mathbb{R}^2}d\boldsymbol{s}\\
&\quad-\int_\Omega\left\langle\nabla(\Phi'(\kappa(u_l)),\frac{\nabla\eta_l}{|\nabla u_l|}-(\nabla u_l\frac{\nabla u_l}{|\nabla u_l|^3})\nabla\eta_l\right\rangle_{\mathbb{R}^2}d\mathbf{x}\\
&= \int_{\partial\Omega}\Phi'(\kappa(u_l))\left\langle\frac{1}{|\nabla u_l|}(\mathbf{I}-(\nabla u_l\frac{\nabla u_l}{|\nabla u_l|^2}))\nabla\eta_l,\boldsymbol{n}\right\rangle_{\mathbb{R}^2}d\boldsymbol{s}\\
&\quad-\int_\Omega\left\langle\nabla(\Phi'(\kappa(u_l))),\frac{1}{|\nabla u_l|}(\mathbf{I}-(\nabla u_l\frac{\nabla u_l}{|\nabla u_l|^2}))\nabla\eta_l\right\rangle_{\mathbb{R}^2}d\mathbf{x}\\
&= \int_{\partial\Omega}\Phi'(\kappa(u_l))\left\langle\frac{1}{|\nabla u_l|}(\mathbf{I}-\mathbf{P})\nabla\eta_l,\boldsymbol{n}\right\rangle_{\mathbb{R}^2}d\boldsymbol{s}\\
&\quad-\int_\Omega\left\langle\frac{1}{|\nabla u_l|}(\mathbf{I}-\mathbf{P})\nabla(\Phi'(\kappa(u_l))),\nabla\eta_l\right\rangle_{\mathbb{R}^2}d\mathbf{x}.
\end{aligned}$$

Here $\mathbf{I}$ is the identity transform and $\mathbf{P} = \nabla u_l\frac{\nabla u_l}{|\nabla u_l|^2} = \vec{\boldsymbol{n}}\otimes\vec{\boldsymbol{n}}$ is the orthogonal projection onto

the normal direction. By using the divergence theorem with the second term, we have

$$\begin{aligned}\delta\overline{\mathcal{R}}^{\mathrm{NewCv}}(u_l;\eta_l) = &\int_{\partial\Omega}\Phi'(\kappa(u_l))\left\langle\frac{1}{|\nabla u_l|}(\mathbf{I}-\mathbf{P})\nabla\eta_l,\boldsymbol{n}\right\rangle_{\mathbb{R}^2}d\boldsymbol{s}\\&+\int_{\Omega}(\nabla\cdot(\frac{1}{|\nabla u_l|}(\mathbf{I}-\mathbf{P})\nabla(\Phi'(\kappa(u_l))))\eta_l d\mathbf{x}\\&-\int_{\partial\Omega}\left\langle\frac{1}{|\nabla u_l|}(\mathbf{I}-\mathbf{P})\nabla(\Phi'(\kappa(u_l)),\boldsymbol{n}\right\rangle_{\mathbb{R}^2}\eta_l d\boldsymbol{s}\end{aligned}$$

or

$$\begin{aligned}\delta\overline{\mathcal{R}}^{\mathrm{NewCv}}(u_l;\eta_l) = &\int_{\Omega}(\nabla\cdot(\frac{1}{|\nabla u_l|}\nabla\Phi'(\kappa(u_l))-\frac{\nabla u_l\cdot\nabla\Phi'(\kappa(u_l))}{(|\nabla u_l|)^3}\nabla u_l))\eta_l d\mathbf{x}\\&+\int_{\partial\Omega}\Phi'(\kappa(u_l))\left\langle\frac{1}{|\nabla u_l|}(\mathbf{I}-\mathbf{P})\nabla\eta_l,\boldsymbol{n}\right\rangle_{\mathbb{R}^2}d\boldsymbol{s}\\&-\int_{\partial\Omega}\left\langle\frac{1}{|\nabla u_l|}(\mathbf{I}-\mathbf{P})\nabla(\Phi'(\kappa(u_l)),\boldsymbol{n}\right\rangle_{\mathbb{R}^2}\eta_l d\boldsymbol{s},\end{aligned}$$

which concludes the proof. ■

Let $\mathcal{R}(\boldsymbol{u}) = \mathcal{R}^{\mathrm{NewCv}}(\boldsymbol{u})$. Then by the sum rule, (3.8), and Theorem 7.2.1, the Euler-Lagrange equations of (7.1) are given by

$$\begin{cases} f_1(\boldsymbol{u})+\alpha\nabla\cdot(\underbrace{\frac{1}{|\nabla u_1|_\beta}\nabla\Phi'(\kappa(u_1))-\frac{\nabla u_1\cdot\nabla\Phi'(\kappa(u_1))}{(|\nabla u_1|_\beta)^3}\nabla u_1}_{\boldsymbol{V}_1=(V_1^1,V_1^2)^\top})=0\\ f_2(\boldsymbol{u})+\alpha\nabla\cdot(\underbrace{\frac{1}{|\nabla u_2|_\beta}\nabla\Phi'(\kappa(u_2))-\frac{\nabla u_2\cdot\nabla\Phi'(\kappa(u_2))}{(|\nabla u_2|_\beta)^3}\nabla u_2}_{\boldsymbol{V}_2=(V_2^1,V_2^2)^\top})=0\end{cases}\quad\text{(new curvature-type model)}\tag{7.15}$$

or in a compact notation

$$\begin{cases} f_1(\boldsymbol{u})+\alpha\nabla\cdot\boldsymbol{V}_1=0,\\ f_2(\boldsymbol{u})+\alpha\nabla\cdot\boldsymbol{V}_2=0,\end{cases}\tag{7.16}$$

subject to the natural boundary conditions $\langle\nabla u_l,\boldsymbol{n}\rangle_{\mathbb{R}^2}=0$ and $\langle\nabla\Phi'(\kappa(u_l)),\boldsymbol{n}\rangle_{\mathbb{R}^2}=0$ on $\partial\Omega$. Here we remark first that the boundary conditions $\langle\nabla u_l,\boldsymbol{n}\rangle_{\mathbb{R}^2}=0$ and $\langle\nabla\Phi'(\kappa(u_l)),\boldsymbol{n}\rangle_{\mathbb{R}^2}=0$ on $\partial\Omega$ are used to drop the first and second boundary integrals in (7.14), respectively. Second we remark that this work mainly considers the case of $\Phi(s)=\frac{1}{2}s^2$ although the general notation allows for other choices.

The proposed regularising functional $\mathcal{R}^{\mathrm{NewCv}}$ has the following properties: i) $\mathcal{R}^{\mathrm{NewCv}}(\mathbf{A}\mathbf{x}+\mathbf{b})=0$ for $\mathbf{A}\in\mathbb{R}^{2\times2}$ and $\mathbf{b}\in\mathbb{R}^2$, i.e $\mathcal{R}^{\mathrm{NewCv}}$ has the same property as the original idea of Fischer–Modersitzki's curvature approach. ii) It preserves discontinuities of $\boldsymbol{u}$ because the diffusion coefficients $\frac{1}{|\nabla u_l|_\beta}\to0$ and $\frac{\nabla u_l\cdot\nabla\Phi'(\kappa(u_l))}{|\nabla u_l|_\beta^3}\to0$ when $|\nabla u_l|_\beta\to\infty$. In other words, for non-smooth deformation problems the new PDE model preserves discontinuities of $\boldsymbol{u}$ by reducing or stopping the diffusion (smoothing) process in inhomogeneous regions presenting large gradients. iii) $\mathcal{R}^{\mathrm{NewCv}}$ reduces to $\mathcal{R}^{\mathrm{FMcurv}}$ in (7.8) if $|\nabla u|\approx0$ and we take $\beta=1$. However $\beta<1$ is required for non-smooth deformation problems. From now on we shall use the notation $\mathcal{R}^{\mathrm{NewCv}}$ to mean *the full curvature model* (7.15) and its numerical solutions is discussed next.

7.3 Numerical solution of the PDE system

While variational models have made many contributions in high-resolution image processing, a major challenge is to implement fast and stable numerical algorithms for solving the associated Euler-Lagrange systems. In this section we briefly review possible numerical methods that have been studied for other models and could be considered for solving (7.16). To proceed, we denote the discrete domain consisting of $N = n^2$ cells of size $h \times h$ by

$$\Omega_h = \{\mathbf{x} \in \Omega | \mathbf{x} = (x_{1_i}, x_{2_j})^\top = ((2i-1)\,h/2, (2j-1)\,h/2),\ 1 \le i, j \le n\}$$

throughout this section where $h = 1/n$ denotes the grid space.

Finite difference discretisation. We shall use a *cell-centered finite difference* approximation for the underlying PDEs. For simplicity, let $(u_l^h)_{i,j} = u_l^h(x_{1_i}, x_{2_j})$ denote the grid functions for $l = 1, 2$. After discretising (7.16), the grid system at $(i, j) \in \Omega_h$ is given by

$$\begin{cases} \underbrace{f_1^h(u_1^h, u_2^h)_{i,j} + \alpha\nabla\cdot(\boldsymbol{V}_1^h)_{i,j}}_{\overline{\mathcal{N}}_1^h(\boldsymbol{u}^h)_{i,j}} = 0 \\ \underbrace{f_2^h(u_1^h, u_2^h)_{i,j} + \nabla\cdot(\boldsymbol{V}_2^h)_{i,j}}_{\overline{\mathcal{N}}_2^h(\boldsymbol{u}^h)_{i,j}} = 0 \end{cases} \quad i.e. \quad \begin{cases} \overline{\mathcal{N}}_1^h(\boldsymbol{u}^h)_{i,j} = 0, \\ \overline{\mathcal{N}}_2^h(\boldsymbol{u}^h)_{i,j} = 0, \end{cases} \tag{7.17}$$

subject to the discrete boundary conditions,

$$\begin{cases} (u_l^h)_{i,1} = (u_l^h)_{i,2}, \quad (u_l^h)_{i,n} = (u_l^h)_{i,n-1}, \quad (u_l^h)_{1,j} = (u_l^h)_{2,j}, \quad (u_l^h)_{n,j} = (u_l^h)_{n-1,j}, \\ \Phi'(\kappa(u_l^h))_{i,1} = \Phi'(\kappa(u_l^h))_{i,2}, \quad \Phi'(\kappa(u_l^h))_{i,n} = \Phi'(\kappa(u_l^h))_{i,n-1}, \\ \Phi'(\kappa(u_l^h))_{1,j} = \Phi'(\kappa(u_l^h))_{2,j}, \quad \Phi'(\kappa(u_l^h))_{n,j} = \Phi'(\kappa(u_l^h))_{n-1,j}, \end{cases} \tag{7.18}$$

with the following notation for the fitting terms f_l from (7.3)

$$\begin{aligned} f_1^h\left(u_1^h, u_2^h\right)_{i,j} &= (I_{i,j}^{h^*} - R_{i,j}^h)((I_{i+1,j}^{h^*} - I_{i-1,j}^{h^*})/\,(2h)), \\ f_2^h\left(u_1^h, u_2^h\right)_{i,j} &= (I_{i,j}^{h^*} - R_{i,j}^h)((I_{i,j+1}^{h^*} - I_{i,j-1}^{h^*})/\,(2h)), \\ I_{i,j}^{h^*} &= I^h(i + \left(u_1^h\right)_{i,j}, j + \left(u_2^h\right)_{i,j}), \\ (\boldsymbol{u}^h)_{i,j} &= ((u_1^h)_{i,j}, (u_2^h)_{i,j})^\top. \end{aligned}$$

Here we approximate the term $\nabla\cdot(\boldsymbol{V}_l^h)_{i,j}$ as follows:

$$(\frac{\partial V_l^1}{\partial x_1})_{i,j} + (\frac{\partial V_l^2}{\partial x_2})_{i,j} = \frac{(V_l^1)_{i+1,j} - (V_l^1)_{i,j}}{h} + \frac{(V_l^2)_{i,j+1} - (V_l^2)_{i,j}}{h}. \tag{7.19}$$

Therefore, we need to calculate V_l^1 at the grid points $(i+1, j)$ and (i, j) and V_l^2 at the grid points $(i, j+1)$ and (i, j). Below we list the approximation used in our numerical realisations

for estimating V_l^1 at the grid point (i,j):

$$\begin{aligned}
\kappa(u_l^h) &= \left[\frac{\delta_{x_1}^-}{h}\left(\frac{\delta_{x_1}^+(u_l^h)_{i,j}/h}{\sqrt{\beta+(\delta_{x_1}^+(u_l^h)_{i,j}/h)^2+(\delta_{x_2}^+(u_l^h)_{i,j}/h)^2}}\right)\right. \\
&\quad \left.+\frac{\delta_{x_2}^-}{h}\left(\frac{\delta_{x_2}^+(u_l^h)_{i,j}/h}{\sqrt{\beta+(\delta_{x_1}^+(u_l^h)_{i,j}/h)^2+(\delta_{x_2}^+(u_l^h)_{i,j}/h)^2}}\right)\right], \\
u_{l_{x_1}}^h &= \delta_{x_1}^+\left(u_l^h\right)_{i,j}/h, \\
u_{l_{x_2}}^h &= \delta_{x_2}^+\left(u_l^h\right)_{i,j}/h, \\
\delta_{x_1}^\pm\left(u_l^h\right)_{i,j} &= \pm\left((u_l^h)_{i\pm1,j}-(u_l^h)_{i,j}\right),\ \delta_{x_2}^\pm\left(u_l^h\right)_{i,j} = \pm\left((u_l^h)_{i,j\pm1}-(u_l^h)_{i,j}\right), \\
|\nabla u_l|_\beta &= \sqrt{\beta+(\delta_{x_1}^+(u_l^h)_{i,j}/h)^2+(\delta_{x_2}^+(u_l^h)_{i,j}/h)^2}, \\
(\Phi'(\kappa(u_l)))_{x_1} &= [\Phi'(\kappa(u_l)_{i+1,j})-\Phi'(\kappa(u_l)_{i,j})]/h, \\
(\Phi'(\kappa(u_l)))_{x_2} &= [\Phi'(\kappa(u_l)_{i,j+1})-\Phi'(\kappa(u_l)_{i,j})]/h.
\end{aligned}$$

Discretisation for V_l^1 at the grid point $(i+1,j)$ and V_l^2 at the grid points $(i,j+1)$ and (i,j) can be given similarly.

7.3.1 Method 1 – A semi-implicit time marching (SITM) method

As discussed in §3.5.1, the main idea of time marching approaches is to introduce an artificial time variable t and compute the steady-state solution of the system of time-dependent PDEs of the form:

$$\begin{cases} \partial_t u_1(\mathbf{x};t)+\overline{\mathcal{N}}_1(\boldsymbol{u}(\mathbf{x};t))=0 \\ \partial_t u_2(\mathbf{x};t)+\overline{\mathcal{N}}_2(\boldsymbol{u}(\mathbf{x};t))=0 \end{cases}$$

where

$$\overline{\mathcal{N}}_l(\boldsymbol{u}(\mathbf{x};t))=f_l(\boldsymbol{u}(\mathbf{x};t))+\alpha\nabla\cdot\left(\frac{1}{|\nabla u_l(\mathbf{x};t)|_\beta}\nabla\Phi'(\kappa(u_l(\mathbf{x};t)))-\frac{\nabla u_l(\mathbf{x};t)\cdot\nabla\Phi'(\kappa(u_l(\mathbf{x};t)))}{|\nabla u_l(\mathbf{x};t)|_\beta^3}\nabla u_l(\mathbf{x};t)\right).$$

In order to overcome the nonlinearity and higher-order derivatives of $\overline{\mathcal{N}}_l$, we linearise the associated system respect to the $(k+1)$th time-step using the method of 'frozen coefficients' and define the iteration step as follows:

$$\begin{cases} (u_1^{(k+1)})_{i,j}=(u_1^{(k)})_{i,j}-\tau\overline{\mathcal{N}}_1^{\text{lin}}(\boldsymbol{u}^{(k+1)})_{i,j} \\ (u_2^{(k+1)})_{i,j}=(u_2^{(k)})_{i,j}-\tau\overline{\mathcal{N}}_2^{\text{lin}}(\boldsymbol{u}^{(k+1)})_{i,j} \end{cases} \quad i.e. \quad \begin{cases} \mathbf{A}_1(u_1^{(k)})(u_1^{(k+1)})_{i,j}=\mathbf{B}_1(\boldsymbol{u}^{(k)})_{i,j}, \\ \mathbf{A}_2(u_2^{(k)})(u_2^{(k+1)})_{i,j}=\mathbf{B}_2(\boldsymbol{u}^{(k)})_{i,j}, \end{cases}$$

or in full details

$$\begin{cases} \underbrace{-\alpha\tau\nabla\cdot\Big(\dfrac{\nabla u_1^{(k)}\cdot\nabla\Phi'(\kappa(u_1^{(k)}))}{|\nabla u_1^{(k)}|_\beta^3}\nabla u_1^{(k+1)}\Big)_{i,j}+(u_1^{(k+1)})_{i,j}=}_{\mathbf{A}_1(u_1^{(k)})(u_1^{(k+1)})_{i,j}} \\ \underbrace{(u_1^{(k)})_{i,j}-\tau[f_1(\boldsymbol{u}^{(k)})_{i,j}+\alpha\nabla\cdot\Big(\dfrac{1}{|\nabla(u_1^{(k)})_{i,j}|_\beta}\nabla\Phi'(\kappa(u_1^{(k)}))_{i,j}]}_{\mathbf{B}_1(\boldsymbol{u}^{(k)})_{i,j}} \\ \underbrace{-\alpha\tau\nabla\cdot\Big(\dfrac{\nabla u_2^{(k)}\cdot\nabla\Phi'(\kappa(u_2^{(k)}))}{|\nabla u_2^{(k)}|_\beta^3}\nabla u_2^{(k+1)}\Big)_{i,j}+(u_2^{(k+1)})_{i,j}=}_{\mathbf{A}_2(u_2^{(k)})(u_2^{(k+1)})_{i,j}} \\ \underbrace{(u_2^{(k)})_{i,j}-\tau[f_2(\boldsymbol{u}^{(k)})_{i,j}+\alpha\nabla\cdot\Big(\dfrac{1}{|\nabla u_2^{(k)}|_\beta}\nabla\Phi'(\kappa(u_2^{(k)}))_{i,j}]}_{\mathbf{B}_2(\boldsymbol{u}^{(k)})_{i,j}} \end{cases} \tag{7.20}$$

which is a semi-implicit time marching scheme for (7.16). Here the symbol h in the previous section is dropped for simplicity and we denote by

$$\overline{\mathcal{N}}_l^{\text{lin}}(\boldsymbol{u}^{(k+1)})_{i,j}=f_l(\boldsymbol{u}^{(k)})_{i,j}+\alpha\nabla\cdot\Big(\frac{1}{|\nabla u_l^{(k)}|_\beta}\nabla\Phi'(\kappa(u_l^{(k)}))-\frac{\nabla u_l^{(k)}\cdot\nabla\Phi'(\kappa(u_l^{(k)}))}{|\nabla u_l^{(k)}|_\beta^3}\nabla u_l^{(k+1)}\Big)_{i,j}. \tag{7.21}$$

the frozen operator, linearised at a grid point (i,j). We note that this frozen operator allows us to solve (7.16) as the system of two second-order PDEs for each time step k because the coefficients from the higher-order derivatives are frozen in the associated discrete system.

7.3.2 Method 2 – A stabilised semi-implicit time marching (SSITM) method

Although this above idea of linearisation via semi-implicitness seems reasonable, we found experimentally that this numerical scheme (7.20) is only stable when τ is small and small τ will lead to slow convergence in the overall registration process. The reason for this stability problem is that the discrete system has a highly nonlinear coefficient $\frac{\nabla u_l^{(k)}\cdot\nabla\Phi'(\kappa(u_l^{(k)}))}{|\nabla u_l^{(k)}|_\beta^3}$ that can easily change its sign for large τ so neither positive-definiteness nor diagonal dominance can be guaranteed for numerical schemes of the underlying system (a matrix form of (7.20))

$$\begin{bmatrix} \mathbf{A}_1(u_1^{(k)}) & 0 \\ 0 & \mathbf{A}_2(u_1^{(k)}) \end{bmatrix}\begin{pmatrix} u_1^{(k+1)} \\ u_2^{(k+1)} \end{pmatrix}=\begin{pmatrix} \mathbf{B}_1(\boldsymbol{u}^{(k)}) \\ \mathbf{B}_2(\boldsymbol{u}^{(k)}) \end{pmatrix}.$$

In order to improve stability, the stabilising terms based on the so-called *convexity-splitting* techniques developed in different contexts ([14, 44, 45]) may be added as follows:

$$\begin{cases} \gamma_1\tau\mathcal{F}(u_1^{(k+1)})_{i,j} - \alpha\tau\nabla\cdot(\frac{\nabla u_1^{(k)}\cdot\nabla\Phi'(\kappa(u_1^{(k)}))}{|\nabla u_1^{(k)}|_\beta^3}\nabla u_1^{(k+1)})_{i,j} + (u_1^{(k+1)})_{i,j} = \gamma_1\tau\mathcal{F}(u_1^{(k)})_{i,j} + (u_{1i,j}^{(k)} \\ \qquad\qquad - \tau[f_1(\boldsymbol{u}^{(k)})_{i,j} + \alpha\nabla\cdot(\frac{1}{|\nabla u_1^{(k)}|_\beta}\nabla\Phi'(\kappa(u_1^{(k)}))_{i,j}] \\ \gamma_2\tau\mathcal{F}(u_2^{(k+1)})_{i,j} - \alpha\tau\nabla\cdot(\frac{\nabla u_2^{(k)}\cdot\nabla\Phi'(\kappa(u_2^{(k)}))}{|\nabla u_2^{(k)}|_\beta^3}\nabla u_2^{(k+1)})_{i,j} + (u_2^{(k+1)})_{i,j} = \gamma_2\tau\mathcal{F}(u_2^{(k)})_{i,j} + (u_2^{(k)})_{i,j} \\ \qquad\qquad - \tau[f_2(\boldsymbol{u}^{(k)})_{i,j} + \alpha\nabla\cdot(\frac{1}{|\nabla u_2^{(k)}|_\beta}\nabla\Phi'(\kappa(u_2^{(k)}))_{i,j}] \end{cases} \tag{7.22}$$

where $\gamma_l > 0$ and $\mathcal{F}(u_l)$ is some appropriate partial differential operator arising from the minimisation of a convex functional, such as $\int_\Omega |\nabla u_l| d\mathbf{x}$ or $\int_\Omega |\nabla u_l|^2 d\mathbf{x}$.

Note that $\frac{1}{|\nabla u_l^{(k)}|_\beta} \to 1$ as $|\nabla u_l^{(k)}| \to 0$ for smooth problems ($\beta = 1$) and $\frac{1}{|\nabla u_l^{(k)}|_\beta} \to 0$ as $|\nabla u_l^{(k)}| \to \infty$ for non-smooth problems ($\beta < 1$). Therefore, $\mathcal{F}(u_l^{(k+1)})_{i,j} = -\nabla\cdot(\frac{\nabla u_l^{(k+1)}}{|\nabla u_l^{(k)}|_\beta})_{i,j}$ smooths $\boldsymbol{u}$ isotropically inside homogeneous regions corresponding to weak gradients and preserves discontinuities of $\boldsymbol{u}$ in inhomogeneous regions representing large gradients by reducing or stopping diffusion process. As a consequence, $\mathcal{F}(u_l^{(k+1)})_{i,j} = -\nabla\cdot(\frac{\nabla u_l^{(k+1)}}{|\nabla u_l^{(k)}|_\beta})_{i,j}$ appears to be an apropriate choice for both smooth and non-smooth registration problems, while $\mathcal{F}(u_l^{(k+1)})_{i,j} = -\Delta(u_l^{(k+1)})_{i,j}$ is only suitable for smooth cases.

7.3.3 Method 3 – Fixed-point (FP) methods

As is well-known [137, 27, 30], fixed-point (FP) methods are usually faster than time marching approaches when appropriate FP schemes are applied. To try this idea, we use a similar linearisation to the above (7.22) plus a linearised version of $f_l(u_1^{[\nu+1]}, u_2^{[\nu+1]})$ via a typical Taylor's expansion as follows

$$\begin{aligned} f_l(u_1^{[\nu+1]}, u_2^{[\nu+1]}) &\approx f_l(u_1^{[\nu]}, u_2^{[\nu]}) + \partial_{u_1} f_l(u_1^{[\nu]}, u_2^{[\nu]})\delta u_1^{[\nu]} + \partial_{u_2} f_l(u_1^{[\nu]}, u_2^{[\nu]})\delta u_2^{[\nu]}, \\ &= f_l(u_1^{[\nu]}, u_2^{[\nu]}) + \sigma_{l1}^{[\nu]}\delta u_1^{[\nu]} + \sigma_{l2}^{[\nu]}\delta u_2^{[\nu]}, \\ &= f_l(u_1^{[\nu]}, u_2^{[\nu]}) + \sigma_{l1}^{[\nu]}(u_1^{[\nu+1]} - u_1^{[\nu]}) + \sigma_{l2}^{[\nu]}(u_2^{[\nu+1]} - u_2^{[\nu]}) \end{aligned} \tag{7.23}$$

where

$$\sigma_{l1}^{[\nu]} = \partial_{u_1} f_l(u_1^{[\nu]}, u_2^{[\nu]}) = (\partial_{u_l} T_{\boldsymbol{u}^{[\nu]}})(\partial_{u_1} T_{\boldsymbol{u}^{[\nu]}}) + (T_{\boldsymbol{u}^{[\nu]}} - R)(\partial_{u_1 u_l} T_{\boldsymbol{u}^{[\nu]}})$$

and

$$\sigma_{l2}^{[\nu]} = \partial_{u_2} f_l(u_1^{[\nu]}, u_2^{[\nu]}) = (\partial_{u_l} T_{\boldsymbol{u}^{[\nu]}})(\partial_{u_2} T_{\boldsymbol{u}^{[\nu]}}) + (T_{\boldsymbol{u}^{[\nu]}} - R)(\partial_{u_2 u_l} T_{\boldsymbol{u}^{[\nu]}}).$$

Then a FP scheme of (7.16) can be given by (for $\nu = 0, 1, 2, 3, ...$)

$$\begin{cases} -\alpha\nabla\cdot(\frac{\nabla u_1^{[\nu]}\cdot\nabla\Phi'(\kappa(u_1^{[\nu]}))}{|\nabla u_1^{[\nu]}|_\beta^3}\nabla u_1^{[\nu+1]})_{i,j} + (\sigma_{11}^{[\nu]})_{i,j}(u_1^{[\nu+1]})_{i,j} + (\sigma_{12}^{[\nu]})_{i,j}(u_2^{[\nu+1]})_{i,j} = \\ \quad (\sigma_{11}^{[\nu]})_{i,j}(u_1^{[\nu]})_{i,j} + (\sigma_{12}^{[\nu]})_{i,j}(u_2^{[\nu]})_{i,j} - f_1(\boldsymbol{u}^{[\nu]})_{i,j} - \alpha\nabla\cdot(\frac{1}{|\nabla u_1^{[\nu]}|_\beta}\nabla\Phi'(\kappa(u_1^{[\nu]})))_{i,j}, \\ -\alpha\nabla\cdot(\frac{\nabla u_2^{[\nu]}\cdot\nabla\Phi'(\kappa(u_2^{[\nu]}))}{|\nabla u_2^{[\nu]}|_\beta^3}\nabla u_2^{[\nu+1]})_{i,j} + (\sigma_{22}^{[\nu]})_{i,j}(u_2^{[\nu+1]})_{i,j} + (\sigma_{21}^{[\nu]}))_{i,j}(u_1^{[\nu+1]})_{i,j} = \\ \quad (\sigma_{21}^{[\nu]})_{i,j}(u_1^{[\nu]})_{i,j} + (\sigma_{22}^{[\nu]})_{i,j}(u_2^{[\nu]})_{i,j} - f_2(\boldsymbol{u}^{[\nu]})_{i,j} - \alpha\nabla\cdot(\frac{1}{|\nabla u_2^{[\nu]}|_\beta}\nabla\Phi'(\kappa(u_2^{[\nu]})))_{i,j}. \end{cases} \tag{7.24}$$

Note that $\sigma_{lm}^{[\nu]}$ $(l,m=1,2)$ can be refined further as mentioned in §5.3.1, i.e. $\sigma_{21}^{[\nu]}=\sigma_{12}^{[\nu]}$ and $\sigma_{lm}^{[\nu]}=(\partial_{u_l}T_{\boldsymbol{u}^{[\nu]}})(\partial_{u_m}T_{\boldsymbol{u}^{[\nu]}})$.

Unfortunately, we found experimentally that this FP scheme is neither *stable* or *convergent*. This difficulty arises from the unbalanced terms in the resulting discrete system. For example, fixing $\beta=10^{-2}$ in the flat regions where $|\nabla u_l|=0$ reduces the diffusion coefficient $\frac{\nabla u_l\cdot\nabla\Phi'(\kappa(u_l))}{|\nabla u_l|_\beta^3}\approx O(10^6)$ compared with that of only $\frac{1}{|\nabla u_l|_\beta}\approx O(10^2)$ for the TV restoration case [118, 137].

7.3.4 Method 4 – A stabilised fixed-point (SFP) method

In order to improve the FP scheme (7.24), the convexity-splitting idea [14, 44, 45] is again considered by adding stabilised terms as follows:

$$\begin{cases}\gamma_1\mathcal{F}(u_1^{[\nu+1]})_{i,j}-\alpha\nabla\cdot(\frac{\nabla u_1^{[\nu]}\cdot\nabla\Phi'(\kappa(u_1^{[\nu]}))}{|\nabla u_1^{[\nu]}|_\beta^3}\nabla u_1^{[\nu+1]})_{i,j}+(\sigma_{11}^{[\nu]})_{i,j}(u_1^{[\nu+1]})_{i,j}+(\sigma_{12}^{[\nu]})_{i,j}(u_2^{[\nu+1]})_{i,j}=\\ \quad\gamma_1\mathcal{F}(u_1^{[\nu]})_{i,j}+(\sigma_{11}^{[\nu]})_{i,j}(u_1^{[\nu]})_{i,j}+(\sigma_{12}^{[\nu]})_{i,j}(u_2^{[\nu]})_{i,j}-f_1(\boldsymbol{u}^{[\nu]})_{i,j},\\ \quad-\alpha\nabla\cdot(\frac{1}{|\nabla u_1^{[\nu]}|_\beta}\nabla\Phi'(\kappa(u_1^{[\nu]})))_{i,j}\\ \\ \gamma_2\mathcal{F}(u_2^{[\nu+1]})_{i,j}-\alpha\nabla\cdot(\frac{\nabla u_2^{[\nu]}\cdot\nabla\Phi'(\kappa(u_2^{[\nu]}))}{|\nabla u_2^{[\nu]}|_\beta^3}\nabla u_2^{[\nu+1]})_{i,j}+(\sigma_{22}^{[\nu]})_{i,j}(u_2^{[\nu+1]})_{i,j}+(\sigma_{21}^{[\nu]}))_{i,j}(u_1^{[\nu+1]})_{i,j}=\\ \quad\gamma_2\mathcal{F}(u_2^{[\nu]})_{i,j}+(\sigma_{21}^{[\nu]})_{i,j}(u_1^{[\nu]})_{i,j}+(\sigma_{22}^{[\nu]})_{i,j}(u_2^{[\nu]})_{i,j}-f_2(\boldsymbol{u}^{[\nu]})_{i,j}\\ \quad-\alpha\nabla\cdot(\frac{1}{|\nabla u_2^{[\nu]}|_\beta}\nabla\Phi'(\kappa(u_2^{[\nu]})))_{i,j}\end{cases}\tag{7.25}$$

and we shall name this resulting FP scheme as the *stabilised fixed-point* (SFP) method.

As mentioned in Method 2, we also found that $\mathcal{F}(u_l^{[\nu+1]})=-\nabla\cdot(\frac{\nabla u_l^{[\nu+1]}}{|\nabla u_l^{[\nu]}|_\beta})$ is a suitable choice for both smooth and non-smooth registration problems. Therefore, our SFP method can be explicitly expressed as follows:

$$\mathbf{N}^{\mathrm{SFP}}[\boldsymbol{u}^{[\nu]}]\boldsymbol{u}^{[\nu+1]}=\mathbf{G}^{\mathrm{SFP}}[\boldsymbol{u}^{[\nu]}]\tag{7.26}$$

where

$$\mathbf{N}^{\mathrm{SFP}}[\boldsymbol{u}^{[\nu]}]=\begin{bmatrix}-\alpha\mathcal{L}_1^{\mathrm{SFP}}[u_1^{[\nu]}]_{i,j} & (\sigma_{12}^{[\nu]})_{i,j}\\ (\sigma_{21}^{[\nu]})_{i,j} & -\alpha\mathcal{L}_2^{\mathrm{SFP}}[u_2^{[\nu]}]_{i,j}\end{bmatrix},\ \mathbf{G}^{\mathrm{SFP}}[\boldsymbol{u}^{[\nu]}]=\begin{pmatrix}(\widehat{g}_1)_{i,j}^{[\nu]}\\ (\widehat{g}_2)_{i,j}^{[\nu]}\end{pmatrix}$$

$$\begin{aligned}(\widehat{g}_l)_{i,j}^{[\nu]}=&-\gamma_l\nabla\cdot(\frac{\nabla u_l^{[\nu]}}{|\nabla u_l^{[\nu]}|_\beta})_{i,j}+(\sigma_{l1}^{[\nu]})_{i,j}(u_1^{[\nu]})_{i,j}+(\sigma_{l2}^{[\nu]})_{i,j}(u_2^{[\nu]})_{i,j}-f_l(\boldsymbol{u}^{[\nu]})_{i,j}\\ &-\alpha\nabla\cdot(\frac{1}{|\nabla u_l^{[\nu]}|_\beta}\nabla\Phi'(\kappa(u_l^{[\nu]})))_{i,j}\end{aligned}$$

and

$$\mathcal{L}_l^{\mathrm{SFP}}[u_l^{[\nu]}]_{i,j}(u_l^{[\nu+1]})_{i,j}=\nabla\cdot(\overbrace{(\frac{\gamma_1}{|\nabla u_l^{[\nu]}|_\beta}+\frac{\nabla u_l^{[\nu]}\cdot\nabla\Phi'(\kappa(u_l^{[\nu]}))}{|\nabla u_l^{[\nu]}|_\beta^3})}^{D_l(u_l^{[\nu]})}\nabla u_l^{[\nu+1]})_{i,j}+(\sigma_{ll}^{[\nu]})_{i,j}(u_l^{[\nu+1]})_{i,j}.$$

In each SFP outer iteration ν, the PCGS relaxation method is used as the inner solver in our numerical scheme to solve approximately the associated linear system. Here the kth PCGS

step is given by

$$(\boldsymbol{u}^{[\nu+1]})_{i,j}^{[k+1]} = \left(\mathbf{N}^{\mathrm{SFP}}[\boldsymbol{u}^{[\nu]}]_{i,j}\right)^{-1} (\mathbf{G}^{\mathrm{SFP}}[\boldsymbol{u}^{[\nu]}])_{i,j}^{[k+1/2]}, \tag{7.27}$$

where

$$\mathbf{N}^{\mathrm{SFP}}[\boldsymbol{u}^{[\nu]}]_{i,j} = \begin{bmatrix} \alpha(\Sigma_1^{[\nu]})_{i,j}/h^2 + (\sigma_{11}^{[\nu]})_{i,j} & (\sigma_{12}^{[\nu]})_{i,j} \\ (\sigma_{21}^{[\nu]})_{i,j} & \alpha(\Sigma_2^{[\nu]})_{i,j}/h^2 + (\sigma_{22}^{[\nu]})_{i,j} \end{bmatrix},$$

$$(\mathbf{G}^{\mathrm{SFP}}[\boldsymbol{u}^{[\nu]}])_{i,j}^{[k+1/2]} = \begin{pmatrix} (\widehat{g}_1)_{i,j}^{[\nu]} + \alpha(\overline{\Sigma}_1^{[\nu]})_{i,j}(u_1^{[\nu+1]})_{i,j}^{[k+1/2]} \\ (\widehat{g}_2)_{i,j}^{[\nu]} + \alpha(\overline{\Sigma}_2^{[\nu]})_{i,j}(u_2^{\nu+1]})_{i,j}^{[k+1/2]} \end{pmatrix},$$

$$(\Sigma_l^{[\nu]})_{i,j} = (2D_{l3}(u_l^{[\nu]})_{i,j} + D_{l1}(u_l^{[\nu]})_{i,j} + D_{l2}(u_l^{[\nu]})_{i,j}),$$

$$\begin{aligned}(\overline{\Sigma}_l^{[\nu]})_{i,j}(u_l^{[\nu+1]})_{i,j}^{[k+1/2]} = (1/h^2)\Big((D_{l3}(u_l^{[\nu]})_{i,j})(u_l^{[\nu+1]})_{i+1,j}^{[k]} + (D_{l1}(u_l^{[\nu]})_{i,j})(u_l^{[\nu+1]})_{i-1,j}^{[k+1]} \\ +(D_{l3}(u_l^{[\nu]})_{i,j})(u_l^{[\nu+1]})_{i,j+1}^{[k]} + (D_{l2}(u_l^{[\nu]})_{i,j})(u_l^{[\nu+1]})_{i,j-1}^{[k+1]}\Big)\end{aligned}$$

$$D_{l1}(u_l^{[\nu]})_{i,j} = D_l(u_l^{[\nu]})_{i-1,j},\ D_{l2}(u_l^{[\nu]})_{i,j} = D_l(u_l^{[\nu]})_{i,j-1},\ D_{l3}(u_l^{[\nu]})_{i,j} = D_l(u_l^{[\nu]})_{i,j}.$$

We remark that other iterative techniques such as the line relaxation techniques or the preconditioned conjugate gradient method may also be used as an inner solver. However, the PCGS relaxation method appears a cheaper option. Finally, we note that the stabilising terms $-\gamma_l \nabla \cdot (\frac{\nabla u_l^{[\nu+1]}}{|\nabla u_l^{[\nu]}|_\beta})$ and $\sigma_{ll}^{[\nu]}$ lead the system (7.26) to be strictly or irreducibly diagonally dominant. This guarantees the existence of a unique solution of each linearised system and global convergence of the Jacobi and GS iterations [117, 121].

7.3.5 Method 5 – A primal-dual fixed-point (PDFP) method

In designing alternative methods for (7.16), we note that the previous four methods tackle the nonlinearity in some way. Below we consider an idea from reducing the higher-order derivatives. In fact, higher-order PDEs (in the context of mixed finite elements or in the denoising model [25]) as well as higher order ordinary differential equations are often reduced to low orders before numerical solution.

In order to apply this idea to (7.16), our first step is to introduce suitable intermediate variables (which we shall call *dual variables*)

$$v_1 = -\Phi'(\kappa(u_1)) = -\nabla \cdot \tfrac{\nabla u_1}{|\nabla u_1|_\beta} \text{ and } v_2 = -\Phi'(\kappa(u_2)) = -\nabla \cdot \tfrac{\nabla u_2}{|\nabla u_2|_\beta},$$

leading to the equivalent system of four second-order nonlinear PDEs given by

$$\begin{cases} -\nabla \cdot \frac{\nabla u_1}{|\nabla u_1|_\beta} - v_1 = g_1 \\ -\nabla \cdot \frac{\nabla u_2}{|\nabla u_2|_\beta} - v_2 = g_2 \\ f_1(\boldsymbol{u}) - \alpha\nabla \cdot (\frac{\nabla v_1}{|\nabla u_1|_\beta} + \frac{\nabla u_1 \cdot (-\nabla v_1)}{|\nabla u_1|_\beta^3}\nabla u_1) = g_3 \\ f_2(\boldsymbol{u}) - \alpha\nabla \cdot (\frac{\nabla v_2}{|\nabla u_2|_\beta} + \frac{\nabla u_2 \cdot (-\nabla v_2)}{|\nabla u_2|_\beta^3}\nabla u_2) = g_4 \end{cases} \tag{7.28}$$

subject to the boundary conditions transferred into $\nabla u_l = 0$ and $\nabla v_l = 0$ for $l = 1, 2$ where $g_{\widehat{l}} = 0$ ($\widehat{l} = 1, \ldots, 4$). The next step is to linearise (7.28) by a FP scheme as follows:

$$\mathbf{N}^{\mathrm{PDFP}}[\boldsymbol{z}^{[\nu]}]\boldsymbol{z}^{[\nu+1]} = \mathbf{G}^{\mathrm{PDFP}}[\boldsymbol{z}^{[\nu]}] \tag{7.29}$$

where linearisation for $f_l(u_1^{[\nu+1]}, u_2^{[\nu+1]})$ is as in (7.23), $\boldsymbol{z}^{[\nu+1]} = (z_1^{[\nu+1]}, z_2^{[\nu+1]}, z_3^{[\nu+1]}, z_4^{[\nu+1]})^\top = (u_1^{[\nu+1]}, u_2^{[\nu+1]}, v_1^{[\nu+1]}, v_2^{[\nu+1]})^\top$,

$$\mathbf{N}^{\mathrm{PDFP}}[\boldsymbol{z}^{[\nu]}] = \begin{bmatrix} -\widetilde{\mathcal{L}}_1[u_1^{[\nu]}] & 0 & -1 & 0 \\ 0 & -\widetilde{\mathcal{L}}_2[u_2^{[\nu]}] & 0 & -1 \\ \sigma_{11}^{[\nu]} & \sigma_{12}^{[\nu]} & -\alpha\widetilde{\mathcal{L}}_1[u_1^{[\nu]}] & 0 \\ \sigma_{21}^{[\nu]} & \sigma_{22}^{[\nu]} & 0 & -\alpha\widetilde{\mathcal{L}}_2[u_2^{[\nu]}] \end{bmatrix},$$

$$\mathbf{G}^{\mathrm{PDFP}}[\boldsymbol{z}^{[\nu]}] = (g_1, g_2, \widehat{g}_3^{[\nu]}, \widehat{g}_4^{[\nu]})^\top,$$

$$\widetilde{\mathcal{L}}_l[u_l^{[\nu]}]z_{\widehat{l}}^{[\nu+1]} = \nabla \cdot (\overbrace{\frac{1}{|\nabla u_l^{[\nu]}|_\beta}}^{\widetilde{D}_l(u_l^{[\nu]})} \nabla z_{\widehat{l}}^{[\nu+1]}) = \ (z_{\widehat{l}}^{[\nu+1]} = u_l^{[\nu+1]} \text{ or } v_l^{[\nu+1]}),$$

$$\widehat{g}_3^{[\nu]} = g_3 - f_1(u_1^{[\nu]}, u_2^{[\nu]}) + \sigma_{11}^{[\nu]}u_1^{[\nu]} + \sigma_{12}^{[\nu]}u_2^{[\nu]} + \alpha\nabla \cdot (\frac{\nabla u_1^{[\nu]} \cdot (-\nabla v_1^{[\nu]})}{|\nabla u_1^{[\nu]}|_\beta^3} \nabla u_1^{[\nu]}),$$

and

$$\widehat{g}_4^{[\nu]} = g_4 - f_2(u_1^{[\nu]}, u_2^{[\nu]}) + \sigma_{22}^{[\nu]}u_2^{[\nu]} + \sigma_{21}^{[\nu]}u_1^{[\nu]} + \alpha\nabla \cdot (\frac{\nabla u_2^{[\nu]} \cdot (-\nabla v_2^{[\nu]})}{|\nabla u_2^{[\nu]}|_\beta^3} \nabla u_2^{[\nu]}).$$

Here discretisation of (7.29) is done as in §7.3. We shall call this numerical scheme by a *primal-dual fixed-point* (PDFP) method because it includes the primal variables u_1, u_2 and the dual variables v_1, v_2 in a FP scheme. We remark that other choices of selecting the dual variables for (7.16) were also tested, but did not work well. For example, introducing the new variables

$$\overrightarrow{v}_1 = \frac{1}{|\nabla u_1|_\beta}\nabla\Phi'(\kappa(u_1)) - \frac{\nabla u_1 \cdot \nabla\Phi'(\kappa(u_1))}{|\nabla u_1|_\beta^3}\nabla u_1$$

and

$$\overrightarrow{v}_2 = \frac{1}{|\nabla u_2|_\beta}\nabla\Phi'(\kappa(u_2)) - \frac{\nabla u_2 \cdot \nabla\Phi'(\kappa(u_2))}{|\nabla u_2|_\beta^3}\nabla u_2$$

can only reduce the resulting PDEs to order three systems. We note further that in our numerical scheme each PDFP outer step is solved using a PCGS relaxation method (as with Method 4 of §7.3.4) as the inner linear solver. Here, such an inner solution step is given by

$$(\boldsymbol{z}^{[\nu+1]})_{i,j}^{[k+1]} = (\mathbf{N}^{\mathrm{PDFP}}[\boldsymbol{z}^{[\nu]}]_{i,j})^{-1}(\mathbf{G}^{\mathrm{PDFP}}[\boldsymbol{z}^{[\nu]}])_{i,j}^{[k+1/2]}, \tag{7.30}$$

where

$$\mathbf{N}^{\mathrm{PDFP}}[\boldsymbol{z}^{[\nu]}]_{i,j} = \begin{bmatrix} (\widetilde{\Sigma}_1^{[\nu]})_{i,j}/h^2 & 0 & -1 & 0 \\ 0 & (\widetilde{\Sigma}_2^{[\nu]})_{i,j}/h^2 & 0 & -1 \\ (\sigma_{11}^{[\nu]})_{i,j} & (\sigma_{12}^{[\nu]})_{i,j} & \alpha(\widetilde{\Sigma}_1^{[\nu]})_{i,j}/h^2 & 0 \\ (\sigma_{21}^{[\nu]})_{i,j} & (\sigma_{22}^{[\nu]})_{i,j} & 0 & \alpha(\widetilde{\Sigma}_2^{[\nu]})_{i,j}/h^2 \end{bmatrix}, \tag{7.31}$$

$$(\mathbf{G}^{\mathrm{PDFP}}[\boldsymbol{z}^{[\nu]}])_{i,j}^{[k+1/2]} = \begin{pmatrix} (g_1)_{i,j} + (\overline{\widetilde{\Sigma}}_1^{[\nu]})_{i,j}(u_1^{[\nu+1]})_{i,j}^{[k+1/2]} \\ (g_2)_{i,j} + (\overline{\widetilde{\Sigma}}_2^{[\nu]})_{i,j}(u_2^{[\nu+1]})_{i,j}^{[k+1/2]} \\ (\widehat{g}_3)_{i,j}^{[\nu]} + \alpha(\overline{\widetilde{\Sigma}}_1^{[\nu]})_{i,j}(v_1^{[\nu+1]})_{i,j}^{[k+1/2]} \\ (\widehat{g}_4)_{i,j}^{[\nu]} + \alpha(\overline{\widetilde{\Sigma}}_2^{[\nu]})_{i,j}(v_2^{[\nu+1]})_{i,j}^{[k+1/2]} \end{pmatrix}, \tag{7.32}$$

$$(\widetilde{\Sigma}_l^{[\nu]})_{i,j} = (2\widetilde{D}_{l3}(u_l^{[\nu]})_{i,j} + \widetilde{D}_{l1}(u_l^{[\nu]})_{i,j} + \widetilde{D}_{l2}(u_l^{[\nu]})_{i,j}), \tag{7.33}$$

and

$$(\widetilde{\overline{\Sigma}}_l^{[\nu]})_{i,j}(z_{\widehat{l}}^{[\nu+1]})_{i,j}^{[k+1/2]} = (1/h^2)\left((\widetilde{D}_{l3}(u_l^{[\nu]})_{i,j})(z_{\widehat{l}}^{[\nu+1]})_{i+1,j}^{[k]} + (\widetilde{D}_{l1}(u_l^{[\nu]})_{i-1,j})(z_{\widehat{l}}^{[\nu+1]})_{i-1,j}^{[k+1]}\right.$$
$$\left.+(\widetilde{D}_{l3}(u_l^{[\nu]})_{i,j})(z_{\widehat{l}}^{[\nu+1]})_{i,j+1}^{[k]} + (\widetilde{D}_{l2}(u_l^{[\nu]})_{i,j-1})(z_{\widehat{l}}^{[\nu+1]})_{i,j-1}^{[k+1]}\right). \quad (7.34)$$

$$\widetilde{D}_{l1}(u_l^{[\nu]})_{i,j} = \widetilde{D}_l(u_l^{[\nu]})_{i-1,j},\ \widetilde{D}_{l2}(u_l^{[\nu]})_{i,j} = \widetilde{D}_l(u_l^{[\nu]})_{i,j-1},\ \widetilde{D}_{l3}(u_l^{[\nu]})_{i,j} = \widetilde{D}_l(u_l^{[\nu]})_{i,j}. \quad (7.35)$$

We note that the approximations in (7.30) – (7.35) need to be adjusted at the image boundary $\partial\Omega_h$ using the homogeneous Neumann boundary conditions i.e.

$$(z_{\widehat{l}}^h)_{i,1} = (z_{\widehat{l}}^h)_{i,2},\ (z_{\widehat{l}}^h)_{i,n} = (z_{\widehat{l}}^h)_{i,n-1},\ (z_{\widehat{l}}^h)_{1,j} = (z_{\widehat{l}}^h)_{2,j},\ (z_{\widehat{l}}^h)_{n,j} = (z_{\widehat{l}}^h)_{n-1,j}. \quad (7.36)$$

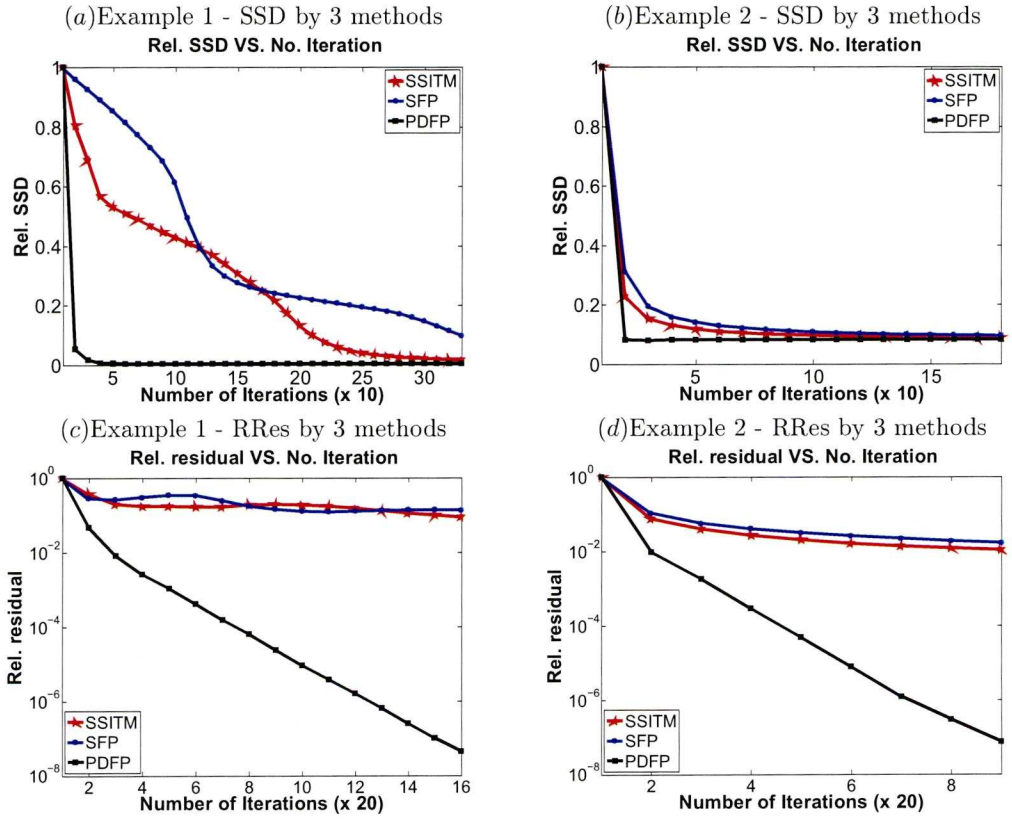
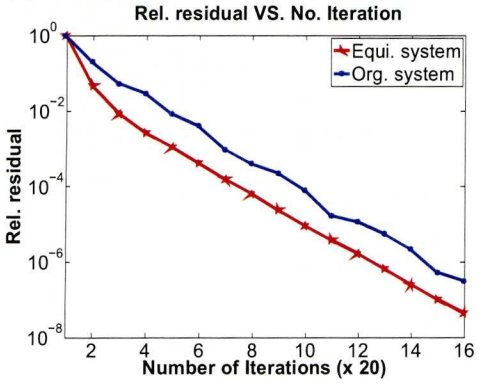


Figure 7.6: Numerical results by Method 2 (SSITM (7.22)), Method 4 (SFP with the FP parameter $\gamma = \gamma_1 = \gamma_2 = 1/\sqrt{\beta}$), and Method 5 (PDFP) for Example 1 (in a 32×32 grid as shown in Figure 7.1 $(a) - (b)$) and Example 2 (as shown in Figure 7.3 $(a) - (b)$). The top two plots show the relative errors in SSD and the bottom plots show the relative residuals versus iterations. Clearly Method 5 (PDFP) performs much better than the other two methods.

We have so far presented five numerical methods for solving (7.16) where Method 2 is enforced by Method 4 and Method 3 is less efficient. So it remains to test the overall performances of the three numerical schemes (i.e. Methods $2, 4, 5$). We tested them for both the smooth Example 2 and the non-smooth Example 1 as respectively shown in Figure 7.3 (a)-(b) and Figure

(e)Example 1 - Old vs New Res by Method 5



(f) Example 2 - Old vs New Res by Method 5

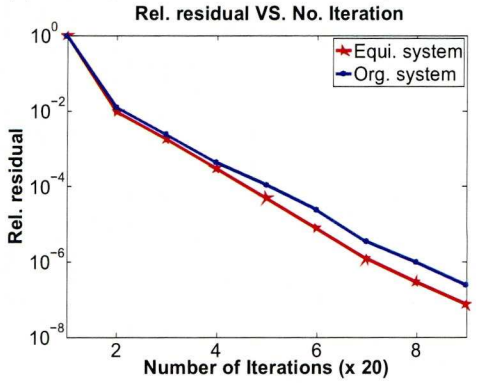


Figure 7.7: Comparison of the relative residuals by Method 5 using both the original system (7.16) and the equivalent system (7.28).

7.1 (a)-(b). The test results shown in Figure 7.6 (a)-(d) show that the new Method 5 performs much better than the others. In particular, as shown Figure 7.7, Method 5 indeed solves the original system (7.16) as expected through an equivalent system.

Although the above tests show that Method 5 is recommended as a unilevel method, our next task is to select a suitable smoother from these methods for designing a convergent MG method for (7.16). To proceed, we shall use a local Fourier analysis to decide which method (4 or 5) is better suited for our purpose. As it turns out, Method 5 is indeed the better method but, even so, modification is still needed for it to be an effective smoother.

7.4 A nonlinear multigrid method

Multigrid techniques [12, 67, 134, 139, 140] have been proved to be very useful in the context of deformable image registration for solving large systems of linear or nonlinear equations arising from high-resolution digital images and real-life applications as briefly reviewed in §3.6.

As is well-known, a working MG has 3 main components: (i) Smoothing via an iterative method; (ii) Restriction from a fine grid to a coarse grid; (iii) Interpolation from a coarse grid to a fine one. On the coarsest grid, an effective unilevel solver is used for accurate solution; here we shall use Method 5. Without reducing the importance of the restriction and interpolation operators, the efficiency of every MG method strongly relies on the efficiency of the *smoother* used at each level. We shall first discuss the choice of smoothers before presenting the overall algorithm.

7.4.1 Local Fourier analysis (LFA)

In this section we shall use the LFA is to analyse the smoothing properties of the proposed smoothers, which are Methods 4 and 5, before considering improvements.

Analysis of Method 4 (Smoother 1)

Here we will compute the smoothing factor of Method 4 iterations (as our Smoother 1 shortly) applied to the linearised system $\mathbf{N}_h^{\mathrm{SFP}}[\overline{\boldsymbol{u}}^h]\boldsymbol{u}^h = \mathbf{G}_h^{\mathrm{SFP}}[\overline{\boldsymbol{u}}^h]$ obtained by freezing coefficients in (7.26) at some outer step. Here $\boldsymbol{u}^h$ and $\overline{\boldsymbol{u}}^h$ denote the exact solution and the current approximation and $\mathbf{N}_h^{\mathrm{SFP}}[\overline{\boldsymbol{u}}^h]$ and $\mathbf{G}_h^{\mathrm{SFP}}[\overline{\boldsymbol{u}}^h]$ the resulting discrete operators from the linearisation at $\overline{\boldsymbol{u}}^h$. The analysis is carried out over the infinite grid

$$\Omega_h^{\infty} = \{\mathbf{x} \in \Omega | \mathbf{x} = (x_{1_i}, x_{2_j})^{\top} = ((2i-1)\,h/2, (2j-1)\,h/2)^{\top},\ i,j \in \mathbb{Z}^2\}. \tag{7.37}$$

Let $\boldsymbol{\varphi}_h(\boldsymbol{\theta}, \mathbf{x}) = \exp(\mathbf{i}\boldsymbol{\theta}\mathbf{x}/h) \cdot \widehat{\mathbf{I}}$ be grid functions, where $\widehat{\mathbf{I}} = (1,1)^{\top}$, $\boldsymbol{\theta} = (\theta_1, \theta_2)^{\top} \in \boldsymbol{\Theta} = (-\pi, \pi]^2$, $\mathbf{x} \in \Omega_h^{\infty}$, and $\mathbf{i} = \sqrt{-1}$. It is important to remark that due to the locality nature of LFA, our analysis applies to each grid point separately i.e., μ is matrix each one of its entries representing the smoothing factor for each grid point $\xi = (i,j)$. Hence we define $\mu_{\mathrm{loc}} = \mu(\xi)$ as the local smoothing factor and $\bar{\mu}_{\mathrm{loc}}$ as the worst possible value of μ_{loc} over Ω_h. Thus for Method 4 from (7.27)

$$\bar{\mu}_{\mathrm{loc}}^{\mathrm{SFP}} = \max_{\xi \in \Omega_h} \mu_{\mathrm{loc}}^{\mathrm{SFP}}$$

To determine $\mu_{\mathrm{loc}}^{\mathrm{SFP}}$ we consider the local discrete system $\mathbf{N}_h^{\mathrm{SFP}}(\xi)\boldsymbol{u}^h(\xi) = \mathbf{G}_h^{\mathrm{SFP}}(\xi)$ centered and defined only within a small neighborhood of ξ and $\boldsymbol{u}^h(\xi) = [u_1^h(\xi), u_2^h(\xi)]$. By using the splitting $\mathbf{N}_h^{\mathrm{SFP}}(\xi) = \mathbf{N}_h^{\mathrm{SFP}+}(\xi) + \mathbf{N}_h^{\mathrm{SFP}-}(\xi)$, it is possible to write the local inner iterations of Method 4 as

$$\mathbf{N}_h^{\mathrm{SFP}+}(\xi)\overline{\boldsymbol{u}}_{new}^h(\xi) + \mathbf{N}_h^{\mathrm{SFP}-}(\xi)\overline{\boldsymbol{u}}_{old}^h(\xi) = \mathbf{G}_h^{\mathrm{SFP}}(\xi) \tag{7.38}$$

where $\overline{\boldsymbol{u}}_{old}^h(\xi)$ and $\overline{\boldsymbol{u}}_{new}^h(\xi)$ stand for the approximations to $\boldsymbol{u}^h(\xi)$ before and after the inner smoothing step, respectively. Here

$$\mathbf{N}_h^{\mathrm{SFP}+}(\xi) = \begin{bmatrix} -\alpha\mathcal{L}_1^{h[+]}(\xi) & \sigma_{12}(\xi) \\ \sigma_{21}(\xi) & -\alpha\mathcal{L}_2^{h[+]}(\xi) \end{bmatrix},\ \mathbf{N}_h^{\mathrm{SFP}-}(\xi) = \begin{bmatrix} -\alpha\mathcal{L}_1^{h[-]}(\xi) & 0 \\ 0 & -\alpha\mathcal{L}_2^{h[-]}(\xi) \end{bmatrix},$$

$$-\mathcal{L}_l^{h[+]}(\xi) = \frac{1}{h^2}\begin{bmatrix} 0 & 0 & 0 \\ -D_{l2}(\overline{u}_l(\xi)) & \Sigma_l(\xi) + (h^2/\alpha)\sigma_{ll}(\xi) & 0 \\ 0 & -D_{l1}(\overline{u}_l(\xi)) & 0 \end{bmatrix},$$

and

$$-\mathcal{L}_l^{h[-]}(\xi) = \frac{1}{h^2}\begin{bmatrix} 0 & -D_{l3}(\overline{u}_l(\xi)) & 0 \\ 0 & 0 & -D_{l3}(\overline{u}_l(\xi)) \\ 0 & 0 & 0 \end{bmatrix}.$$

By subtracting (7.38) from $\mathbf{N}_h^{\mathrm{SFP}}(\xi)\boldsymbol{u}^h(\xi) = \mathbf{G}_h^{\mathrm{SFP}}(\xi)$ and defining $\overline{\boldsymbol{e}}_{new}^h(\xi) = \boldsymbol{u}^h(\xi) - \boldsymbol{u}_{new}^h(\xi)$ and $\overline{\boldsymbol{e}}_{old}^h(\xi) = \boldsymbol{u}^h(\xi) - \boldsymbol{u}_{old}^h(\xi)$ we obtain the local system of error equations

$$\mathbf{N}_h^{\mathrm{SFP}+}(\xi)\overline{\boldsymbol{e}}_{new}^h(\xi) + \mathbf{N}_h^{\mathrm{SFP}-}(\xi)\overline{\boldsymbol{e}}_{old}^h(\xi) = 0 \tag{7.39}$$

or

$$\overline{\boldsymbol{e}}_{new}^h(\xi) = \mathbf{S}_h^{\mathrm{SFP}}(\xi)\overline{\boldsymbol{e}}_{old}^h(\xi) \tag{7.40}$$

where

$$\mathbf{S}_h^{\mathrm{SFP}}(\xi) = -\left[\mathbf{N}_h^{\mathrm{SFP}+}(\xi)\right]^{-1}\left[\mathbf{N}_h^{\mathrm{SFP}-}(\xi)\right]$$

is the amplification factor. The effect of $\mathbf{S}_h^{\mathrm{SFP}}(\xi)$ on the grid functions $\boldsymbol{\varphi}_h(\boldsymbol{\theta}, \mathbf{x})$ within $\boldsymbol{\Theta}_{\mathrm{high}} = \boldsymbol{\Theta} \backslash [-\pi/2, \pi/2]^2$ will determine the smoothing properties of Method 4. Thus, $-\mathcal{L}_l^{h[+]}(\xi)$ and $-\mathcal{L}_l^{h[-]}(\xi)$ are defined in the Fourier modes by

$$-\mathcal{L}_l^{h[+]}(\xi, \boldsymbol{\theta}) = \frac{1}{h^2}(\Sigma_l(\xi) + (h^2/\alpha)\sigma_{ll}(\xi) - D_{l1}(\xi)\exp(-\mathbf{i}\theta_1) - D_{l2}(\xi)\exp(-\mathbf{i}\theta_2)))$$

and

$$-\mathcal{L}_l^{h[-]}(\xi, \boldsymbol{\theta}) = -\frac{1}{h^2}(D_{l3}(\xi)(\exp(\mathbf{i}\theta_1) + \exp(\mathbf{i}\theta_2)))$$

and the local smoothing factor is

$$\mu_{\mathrm{loc}}^{\mathrm{SFP}} = \sup\{|\rho(\mathbf{S}_h^{\mathrm{SFP}}(\xi, \boldsymbol{\theta}))| : \boldsymbol{\theta} \in \boldsymbol{\Theta}_{\mathrm{high}}\} \tag{7.41}$$

where ρ indicates the spectral radius of

$$\mathbf{S}_h^{\mathrm{SFP}}(\xi, \boldsymbol{\theta}) = -\left[\mathbf{N}_h^{\mathrm{SFP}+}(\xi, \boldsymbol{\theta})\right]^{-1}\left[\mathbf{N}_h^{\mathrm{SFP}-}(\xi, \boldsymbol{\theta})\right].$$

On a discrete grid of $[-\pi/2, \pi/2]$, we shall be able to estimate the above factor shortly.

Analysis of Method 5 (Smoother 2)

Now we consider the smoothing factor of Method 5 from (7.29). To this end $\mathbf{N}_h^{\mathrm{PDFP}}[\overline{\boldsymbol{z}}^h]\boldsymbol{z}^h = \mathbf{G}_h^{\mathrm{PDFP}}[\overline{\boldsymbol{z}}^h]$ will denote the linearised system with $\boldsymbol{z}^h$ and $\overline{\boldsymbol{z}}^h$ the exact solution and current approximation. Here the grid function is defined by $\boldsymbol{\varphi}_h(\boldsymbol{\theta}, \mathbf{x}) = \exp(\mathbf{i}\boldsymbol{\theta}\mathbf{x}/h) \cdot \widehat{\mathbf{I}}$, where $\widehat{\mathbf{I}} = (1, 1, 1, 1)^\top$. The local inner iterations for the PDFP algorithm can therefore be written as

$$\mathbf{N}_h^{\mathrm{PDFP}+}(\xi)\overline{\boldsymbol{z}}_{new}^h(\xi) + \mathbf{N}_h^{\mathrm{PDFP}-}(\xi)\overline{\boldsymbol{z}}_{old}^h(\xi) = \mathbf{G}_h^{\mathrm{PDFP}}(\xi) \tag{7.42}$$

where

$$\mathbf{N}_h^{\mathrm{PDFP}+}(\xi) = \begin{bmatrix} -\widetilde{\mathcal{L}}_1^{h[+]}(\xi) & 0 & -1 & 0 \\ 0 & -\widetilde{\mathcal{L}}_2^{h[+]}(\xi) & 0 & -1 \\ \sigma_{11}(\xi) & \sigma_{12}(\xi) & -\alpha\widetilde{\mathcal{L}}_1^{h[+]}(\xi) & 0 \\ \sigma_{21}(\xi) & \sigma_{22}(\xi) & 0 & -\alpha\widetilde{\mathcal{L}}_2^{h[+]}(\xi) \end{bmatrix},$$

$$\mathbf{N}_h^{\mathrm{PDFP}-}(\xi) = \begin{bmatrix} -\widetilde{\mathcal{L}}_1^{h[-]}(\xi) & 0 & 0 & 0 \\ 0 & -\widetilde{\mathcal{L}}_2^{h[-]}(\xi) & 0 & 0 \\ 0 & 0 & -\alpha\widetilde{\mathcal{L}}_1^{h[-]}(\xi) & 0 \\ 0 & 0 & 0 & -\alpha\widetilde{\mathcal{L}}_2^{h[-]}(\xi) \end{bmatrix},$$

$$-\widetilde{\mathcal{L}}_l^{h[+]}(\xi) = \frac{1}{h^2}\begin{bmatrix} 0 & 0 & 0 \\ -\widetilde{D}_{l2}(\overline{u}_l(\xi)) & \widetilde{\Sigma}_l(\xi) & 0 \\ 0 & -\widetilde{D}_{l1}(\overline{u}_l(\xi)) & 0 \end{bmatrix},$$

and

$$-\widetilde{\mathcal{L}}_l^{h[-]}(\xi) = \frac{1}{h^2}\begin{bmatrix} 0 & -\widetilde{D}_{l3}(\overline{u}_l(\xi)) & 0 \\ 0 & 0 & -\widetilde{D}_{l3}(\overline{u}_l(\xi)) \\ 0 & 0 & 0 \end{bmatrix}.$$

Following the similar process of subtracting (7.42) from $\mathbf{N}_h^{\mathrm{PDFP}}(\xi)\boldsymbol{z}^h(\xi) = \mathbf{G}_h^{\mathrm{PDFP}}(\xi)$ one obtains the system of local error equations $\mathbf{N}_h^{\mathrm{PDFP}+}(\xi)\overline{\boldsymbol{e}}_{new}^h(\xi) + \mathbf{N}_h^{\mathrm{PDFP}-}(\xi)\overline{\boldsymbol{e}}_{old}^h(\xi) = 0$ or $\overline{\boldsymbol{e}}_{new}^h(\xi) = \mathbf{S}_h^{\mathrm{PDFP}}(\xi)\overline{\boldsymbol{e}}_{old}^h(\xi)$ where $\overline{\boldsymbol{e}}_{old}^h(\xi) = \boldsymbol{z}^h(\xi) - \overline{\boldsymbol{z}}_{old}^h(\xi)$ and $\overline{\boldsymbol{e}}_{new}^h(\xi) = \boldsymbol{z}^h(\xi) - \overline{\boldsymbol{z}}_{new}^h(\xi)$ are

the error functions and $\mathbf{S}_h^{\rm PDFP}(\xi) = -\left[\mathbf{N}_h^{\rm PDFP+}(\xi)\right]^{-1}\left[\mathbf{N}_h^{\rm PDFP-}(\xi)\right]$. Hence, by considering the grid functions $\boldsymbol{\varphi}_h(\boldsymbol{\theta},\mathbf{x})$, we can represent $\widetilde{\mathcal{L}}_l^{h[+]}(\xi,\boldsymbol{\theta})$ and $\widetilde{\mathcal{L}}_l^{h[+]}(\xi,\boldsymbol{\theta})$ in the Fourier modes by

$$-\widetilde{\mathcal{L}}_l^{h[+]}(\xi,\boldsymbol{\theta}) = \frac{1}{h^2}(\widetilde{\Sigma}_l(\xi) - \widetilde{D}_{l1}(\xi)\exp(-\mathbf{i}\theta_1) - \widetilde{D}_{l2}(\xi)\exp(-\mathbf{i}\theta_2))$$

and

$$-\widetilde{\mathcal{L}}_l^{h[+]}(\xi,\boldsymbol{\theta}) = -\frac{1}{h^2}(\widetilde{D}_{l3}(\xi)(\exp(\mathbf{i}\theta_1) + \exp(\mathbf{i}\theta_2))).$$

From here, the PDFP local smoothing factor is defined by

$$\mu_{\rm loc}^{\rm PDFP} = \sup\{|\rho(\mathbf{S}_h^{\rm PDFP}(\xi,\boldsymbol{\theta}))| : \boldsymbol{\theta}\in\boldsymbol{\Theta}_{\rm high}\}. \tag{7.43}$$

The effectiveness of the above 2 smoothers (i.e. Methods 4, 5) is now tested by computing their smooth rates for Examples $1-2$. The following Table 7.1 summarises the smoothing factors of Smoother 1 (SFP) and Smoother 2 (PDFP) for Examples $1-2$. Clearly for the

Smoother	**Example 1** (non-smooth)	**Example 2** (smooth)
1	0.9410	0.6825
2	0.9412	0.5212

Table 7.1: Smoothing factors $\mu_{\rm loc}$ after 10 outer iterations with $PCGSiter = 10$ by the SFP- and PDFP-type smoothers for the smooth and non-smooth registration problems in Examples $1-2$ as shown respectively in Figure 7.1 $(a)-(b)$ and 7.3 (a)-(b).

smooth Example 2, both Smoothers 1 and 2 are effective and in particular Smoother 2 is better than Smoother 1. But for the non-smooth Example 1, they are much less efficient. Next we consider a method to improve the smoothers and primarily to improve Smoother 2.

7.4.2 A new smoother and its analysis (Smoother 2*)

Recall that μ is a matrix of amplification factors, whose maximum defines the smoothing factor as in (7.41) and (7.43). It turns out that the largest entries of μ coincide with locations where we observe strong jumps of the diffusion coefficients $\widetilde{D}_{l*}(\xi)$. Therefore, our idea of modifying the smoothers is to seek alternative ways to update the solutions where the diffusion coefficients have large jumps. Denote by set W all those pixels with such large coefficients jumps. The whole domain $\Omega_h = W \cup (\Omega_h \backslash W)$ admits two different iterative solvers.

We consider an under-relaxation idea for the sub-domain W (representing the jumps of $\widetilde{D}_l(\xi)$) by updating all these odd points by

$$(\boldsymbol{z}^{[\nu+1]})_{i,j}^{[k+1]} = (1-\omega)(\boldsymbol{z}^{[\nu+1]})_{i,j}^{[k]} + \omega\underbrace{(\mathbf{N}^{\rm PDFP}[\boldsymbol{z}^{[\nu]}]_{i,j})^{-1}(\mathbf{G}^{\rm PDFP}[\boldsymbol{z}^{[\nu]}])_{i,j}^{[k+1/2]}}_{\text{Standard PCGS step}} \tag{7.44}$$

where ω is to be chosen next. As with the previous section, we can analyse the smoothing factor for the $\omega-$PCGS relaxation method in (7.44) by the LFA in the similar way to (7.30). Here

$$-\widetilde{\mathcal{L}}_l^{h[+]}(\xi,\boldsymbol{\theta}) \quad \text{and} \quad -\widetilde{\mathcal{L}}_l^{h[-]}(\xi,\boldsymbol{\theta})$$

are given by

$$-\widetilde{\mathcal{L}}_l^{h[+]}(\xi,\boldsymbol{\theta}) = \frac{1}{h^2}(\widetilde{\Sigma}_l(\xi) - \omega\widetilde{D}_{l1}(\xi)\exp(-\mathbf{i}\theta_1) - \omega\widetilde{D}_{l2}(\xi)\exp(-\mathbf{i}\theta_2)) \tag{7.45}$$

and

$$-\widetilde{\mathcal{L}}_l^{h[-]}(\xi,\boldsymbol{\theta}) = \frac{1}{h^2}((1-\omega)\widetilde{\Sigma}_l(\xi) - \omega\widetilde{D}_{l3}(\xi)(\exp(\mathbf{i}\theta_1) + \exp(\mathbf{i}\theta_2))). \tag{7.46}$$

Further with the updated formulae for $\widetilde{\mathcal{L}}_l^{h[+]}, \widetilde{\mathcal{L}}_l^{h[-]}$ at set W, the amplification factor $\rho(\mathbf{S}_h^{\text{PDFP II}})$ is similarly defined using the updated formulae for $\mathbf{S}_h^{\text{PDFP II}}(\xi) = -\left[\mathbf{N}_h^{\text{PDFP}+}(\xi)\right]^{-1}\left[\mathbf{N}_h^{\text{PDFP}-}(\xi)\right]$. Finally the overall smoothing factor is

$$\mu_{\text{loc}}^{\text{PDFP II}} = \max\left\{\sup_{\xi\in W}\{|\rho(\mathbf{S}_h^{\text{PDFP II}}(\xi,\boldsymbol{\theta}))| : \boldsymbol{\theta}\in\boldsymbol{\Theta}_{\text{high}}\}, \sup_{\xi\in\Omega_h\setminus W}\{|\rho(\mathbf{S}_h^{\text{PDFP}}(\xi,\boldsymbol{\theta}))| : \boldsymbol{\theta}\in\boldsymbol{\Theta}_{\text{high}}\}\right\}.$$

For completeness, we also applied this idea of introducing ω in W for Smoother 1 (SFP from (7.27)) and did a similar LFA analysis. For the same test examples as with Table 7.1, we now show the improved smoothing rates computed for the modified smoothers in Table 7.2 where we name the modified Smoother 2 (i.e. Method 5, PDFP II) as Smoother 2* and the modified Smoother 1 (i.e. from Method 4, SFP II) as Smoother 1*. Clearly we see that the above under-relaxation idea does help improve Smoothers 1, 2; since more improvement can be observed in Smoother 2* (PDFP II) over Smoother 2 (PDFP), we shall take Smoother 2* as our recommended smoother.

Smoother	**Example 1** (non-smooth)	**Example 2** (smooth)
1* ($\omega = 0.7$)	0.8324	0.6711
2* ($\omega = 0.7$)	0.7613	0.5210

Table 7.2: Improved smoothing factors μ_{loc} after using ω under-relaxation idea in sub-domain W Examples 1 – 2.

7.4.3 Nonlinear multigrid algorithm

FAS-NMG method has become an efficient approach for solving nonlinear problems, in particular image processing applications. Here instead of a scalar PDE we have a coupled system of four second-order nonlinear PDEs from (7.28):

$$\mathcal{N}(\boldsymbol{u}^h) = \boldsymbol{g}^h, \quad i.e. \quad \begin{cases} \mathcal{N}_1^h(\boldsymbol{u}^h) = g_1^h \\ \vdots \\ \mathcal{N}_4^h(\boldsymbol{u}^h) = g_4^h \end{cases}$$

involving the nonlinear partial differential operator $\mathcal{N}_{\widehat{l}}^h(\boldsymbol{u}^h)$ given by the left-hand side of (7.28), where $g_{\widehat{l}} = 0$ on the finest grid, for $\widehat{l} = 1, \ldots, 4$.

In our FAS-NMG framework for solving (7.16) via (7.28). Standard coarsening is used first in computing the coarse-grid domain Ω_H by doubling the grid size in each space direction, i.e. $h \to 2h = H$. Second for intergrid transfer operators between Ω_h and Ω_H, the averaging and bi-linear interpolation techniques are used for the restriction and interpolation operators

denoted respectively by I_h^H and I_H^h. In order to compute the coarse-grid operator of $\mathcal{N}_l^h(\boldsymbol{u}^h)$ consisting of two main parts: $f_l^h(u_1^h, u_2^h)$ and $\widetilde{\mathcal{L}}_l^h(u_l^h)$, the DCA method is used.

Below, we present our recommended Smoother 2* (modified Method 5) as an algorithm before presentation of the overall algorithm for solving (7.16).

Algorithm 7.4.1 (Recommended Smoother 2* (PDFP II))

Denote by

α regularisation parameter

ω relaxation parameter

$K > 0$ tolerance (typically $K = 10$)

$PCGSiter$ the maximum number of PCGS iterations

$$[\overline{\boldsymbol{z}}^h] \leftarrow Smoother\left(\overline{\boldsymbol{z}}^h, g_1^h, g_2^h, g_3^h, g_4^h, R^h, T^h, \alpha, \omega, K, PCGSiter\right)$$

- Use input parameters to compute $(\sigma_{lm})_{i,j}$, $(\mathbf{G}_h^{\mathrm{PDFP}}[\overline{\boldsymbol{z}}^h])_{i,j}$, and $(\mathbf{N}^{h\,\mathrm{PDFP}}[\overline{\boldsymbol{z}}^h]_{i,j})^{-1}$ for $l, m = 1, 2$ and $1 \le i, j \le n$
- Perform PCGS steps
  - for $k = 1 : PCGSiter$
    - for $i = 1 : n$
      - for $j = 1 : n$
        - if $\widetilde{D}_l(\overline{u}_l)_{i,j} \ge K \cdot mean\{\widetilde{D}_{l1}(\overline{u}_l)_{i,j}, \widetilde{D}_{l2}(\overline{u}_l)_{i,j}, \widetilde{D}_3(\overline{u}_l)_{i,j}\}$ for $l = 1$ or 2
          - Set $\omega = 0.7$

          else
          - Set $\omega = 0.0$

          end
        - Compute$(\widetilde{\boldsymbol{z}}^h)_{i,j}^{[k+1]}$ using (7.30)
        - $(\overline{\boldsymbol{z}})_{i,j}^{[k+1]} = (1-\omega)(\overline{\boldsymbol{z}})_{i,j}^{[k]} + \omega(\widetilde{\boldsymbol{z}}^h)_{i,j}^{[k+1]}$
      - end
    - end
  - end

To solve (7.28) numerically, our FAS-NMG method with the proposed MG smoother given by Algorithm 7.4.1 is applied recursively down to the coarsest grid consisting of a small number of grid points, typically 8×8. A pseudo-code implementation of our FAS multigrid method is then summarised in the following algorithm:

Algorithm 7.4.2 (FAS-NMG Algorithm)

Denote FAS-NMG parameters as follows:

ν_1 pre-smoothing steps on each level

ν_2 post-smoothing steps on each level

μ the number of multigrid cycles on each level (μ= 1 for V−cycling and $\mu = 2$ for W−cycling). [Here we present the V−cycle with μ= 1.]

α regularisation parameter

ω relaxation parameter

$K > 0$ tolerance

$PCGSiter$ the maximum number of iterations using a smoother

$$\overline{\boldsymbol{z}}^h \leftarrow FASMG\left(\overline{\boldsymbol{z}}^h, \alpha, \overrightarrow{\varepsilon}\right)$$

- Select α, $\overrightarrow{\varepsilon} = (\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4)$ and initial guess solutions $\overline{\boldsymbol{z}}^h_{initial} = (\overline{z}^h_1, \overline{z}^h_2, \overline{z}^h_3, \overline{z}^h_4)^\top$ on the finest grid
- Set $K = 0$, $[\boldsymbol{z}^h]^K = \overline{\boldsymbol{z}}^h_{initial}$, $\widetilde{\varepsilon}_2 = \varepsilon_2 + 1$, $\widetilde{\varepsilon}_3 = \varepsilon_3 + 1$, and $\widetilde{\varepsilon}_4 = \varepsilon_4 + 1$
- While ($K < \varepsilon_1$ AND $\widetilde{\varepsilon}_2 > \varepsilon_2$ AND $\widetilde{\varepsilon}_3 > \varepsilon_3$ AND $\widetilde{\varepsilon}_4 > \varepsilon_4$)
  - $[\overline{\boldsymbol{z}}^h]^{K+1} \leftarrow FASCYC(\overline{\boldsymbol{z}}^h, g^h_1, g^h_2, g^h_3, g^h_4, R^h, T^h, \nu_1, \nu_2, \alpha, \omega, PCGSiter)$
  - $\widetilde{\varepsilon}_2 = mean\{||g^h_l - \mathcal{N}^h_l((\overline{\boldsymbol{z}}^h)^{K+1})||_2 / ||g^h_l - \mathcal{N}^h_l(\overline{\boldsymbol{z}}^h_{initial})||_2 \mid l = 1, 4\}$
  - $\widetilde{\varepsilon}_3 = \mathcal{D}^h(R^h, T^h_{(\overline{\boldsymbol{u}}^h)^{K+1}}) / \mathcal{D}^h(R^h, T^h)$, [Recall that $\mathcal{D}^h\left(R^h, T^h_{(\cdot)}\right) \sim \frac{h^2}{2}||R^h, T^h_{(\cdot)}||^2_2$]
  - $\widetilde{\varepsilon}_4 = |\mathcal{D}^h(R^h, T^h_{(\overline{\boldsymbol{u}}^h)^{K+1}}) - \mathcal{D}^h(R^h, T^h_{(\overline{\boldsymbol{u}}^h)^K})|$
  - $K = K + 1$
- end

---

where

$$\left[\overline{\boldsymbol{z}}^h\right] \leftarrow FASCYC(\overline{\boldsymbol{z}}^h, g^h_1, g^h_2, g^h_3, g^h_4, R^h, T^h, \nu_1, \nu_2, \alpha, \omega, K, PCGSiter)$$

- If $\Omega_h = coarset\ grid$ ($|\Omega_h| = 8 \times 8$), solve (7.28) using Algorithm 7.4.1 and then stop. Else continue with following step.
- Pre-smoothing:
  For $k = 1$ to ν_1, $\left[\overline{\boldsymbol{z}}^h\right] \leftarrow Smoother(\overline{\boldsymbol{z}}^h, g^h_1, g^h_2, g^h_3, g^h_4, R^h, T^h, \alpha, \omega, K, PCGSiter)$
- Restriction to the coarse grid:
  $\overline{z}^H_1 \leftarrow I^H_h \overline{z}^h_1$, $\overline{z}^H_2 \leftarrow I^H_h \overline{z}^h_2$, $\overline{z}^H_3 \leftarrow I^H_h \overline{z}^h_3$, $\overline{z}^H_4 \leftarrow I^H_h \overline{z}^h_4$, $R^H \leftarrow I^H_h R^h$, $T^H \leftarrow I^H_h T^h$
- Set the initial solution for the coarse-grid problem:
  $\left[\widetilde{z}^H_1, \widetilde{z}^H_2, \widetilde{z}^H_3, \widetilde{z}^H_4\right] \leftarrow \left[\overline{z}^H_1, \overline{z}^H_2, \overline{z}^H_3, \overline{z}^H_4\right]$
- Compute the new right-hand side for the coarse-grid problem:
  $g^H_1 \leftarrow I^H_h(g^h_1 - \mathcal{N}^h_1(\overline{\boldsymbol{z}}^h)) + \mathcal{N}^H_1\left(\overline{\boldsymbol{z}}^H\right)$, $g^H_2 \leftarrow I^H_h(g^h_2 - \mathcal{N}^h_2\left(\overline{\boldsymbol{z}}^h\right)) + \mathcal{N}^H_2(\overline{\boldsymbol{z}}^H)$,
  $g^H_3 \leftarrow I^H_h(g^h_3 - \mathcal{N}^h_3(\overline{\boldsymbol{z}}^h)) + \mathcal{N}^H_3\left(\overline{\boldsymbol{z}}^H\right)$, $g^H_4 \leftarrow I^H_h(g^h_4 - \mathcal{N}^h_4(\overline{\boldsymbol{z}}^h)) + \mathcal{N}^H_4(\overline{\boldsymbol{z}}^H)$
- Implement the FAS multigrid on the coarse-grid problem:
  For $k = 1$ toμ, $\left[\overline{\boldsymbol{z}}^H\right] \leftarrow FASCYC\left(\overline{\boldsymbol{z}}^H, g^H_1, g^H_2, g^H_3, g^H_4, R^H, T^H, \nu_1, \nu_2, \alpha, \omega, K, PCGSiter\right)$
- Add the coarse-grid corrections:
  $\overline{z}^h_1 \leftarrow \overline{z}^h_1 + I^h_H\left(\overline{z}^H_1 - \widetilde{z}^H_1\right)$, $\overline{z}^h_2 \leftarrow \overline{z}^h_2 + I^h_H\left(\overline{z}^H_2 - \widetilde{z}^H_2\right)$
  $\overline{z}^h_3 \leftarrow \overline{z}^h_3 + I^h_H\left(\overline{z}^H_3 - \widetilde{z}^H_3\right)$, $\overline{z}^h_4 \leftarrow \overline{z}^h_4 + I^h_H\left(\overline{z}^H_4 - \widetilde{z}^H_4\right)$
- Post-smoothing:
  For $k = 1$ to ν_2, $\left[\overline{\boldsymbol{z}}^h\right] \leftarrow Smoother(\overline{\boldsymbol{z}}^h, g^h_1, g^h_2, g^h_3, g^h_4, R^h, T^h, \alpha, \omega, K, PCGSiter)$

For practical applications our FAS-NMG method is stopped if the maximum number of V− or W−cycles ε_1 is reached (usually $\varepsilon_1 = 10$), the mean of the relative residuals obtained from the Euler-Lagrange equations (7.28) is smaller than a small number $\varepsilon_2 > 0$ (typically $\varepsilon_2 = 10^{-3}$), the relative reduction of the dissimilarity is smaller than some $\varepsilon_3 > 0$ (we usually assign $\varepsilon_3 = 0.3$ meaning that the relative reduction of the dissimilarity would decrease about 70%), or the change in two consecutive steps of the data/fitting term $\mathcal{D}$ is smaller than a small number $\varepsilon_4 > 0$ (typically $\varepsilon_4 = 10^{-6}$).

7.5 Further numerical experiments

In this section some experiments are provided to

(i) compare the modelling results of our new curvature model $\mathcal{R}^{\mathrm{NewCv}}$ with two related approximation models $\mathcal{R}^{\mathrm{FMcurv}}$ and $\mathcal{R}^{\mathrm{HWcurv}}$;

(ii) demonstrate the performance of our new FAS-NMG algorithm for $\mathcal{R}^{\text{NewCv}}$ with regard to parameter changes.

Note that our FAS-NMG algorithm also works for the models of $\mathcal{R}^{\text{FMcurv}}$ and $\mathcal{R}^{\text{HWcurv}}$. Two typical data sets (a smooth registration problem and a non-smooth registration problem to be denoted respectively as Example 3[1] and Example 4) are selected for the experiments, as shown respectively in Figure 7.8 $(a) - (d)$. Improvements of $\mathcal{R}^{\text{FMcurv}}$ and $\mathcal{R}^{\text{HWcurv}}$ over non-curvature models can also be found from [47, 48, 79, 78, 73, 75, 74]. In all cases, we use the bilinear interpolation to compute the transformed template image $T_{\boldsymbol{u}}$ once the displacement field is found. Below we mainly highlight the further gains from using $\mathcal{R}^{\text{NewCv}}$.

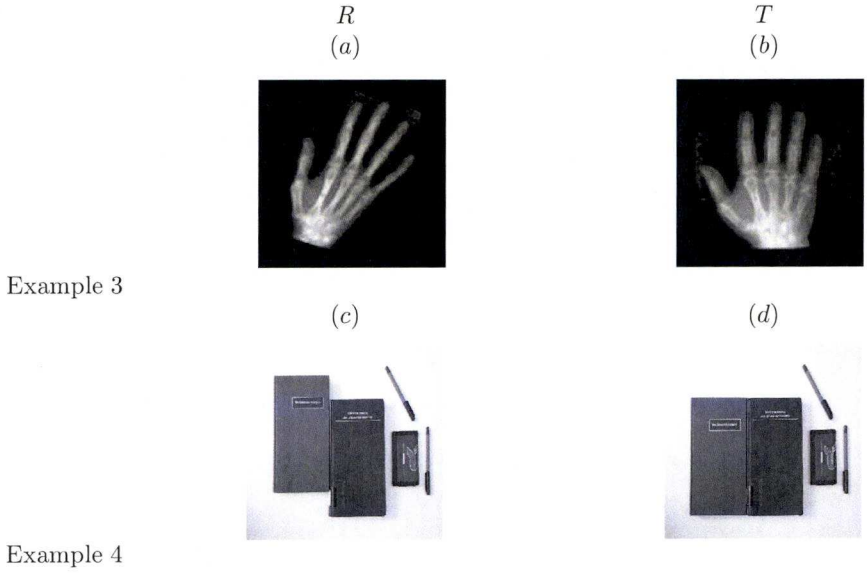


Figure 7.8: The second set of 2 registration problems. Left to right: reference R and template T. Top to bottom: Example 3 (a smooth registration problem) and Example 4 (a non-smooth registration problem).

7.5.1 Comparison with other PDE-based image registration models

In the first experiment, our aim is to investigate capabilities of $\mathcal{R}^{\text{FMcurv}}$, $\mathcal{R}^{\text{HWcurv}}$, and $\mathcal{R}^{\text{NewCv}}$ for registration of the two test Examples $3 - 4$ in resolution 512×512.

The registered results by the three models are shown in Figure 7.9 $(a) - (f)$ with the deformation results shown in Figures 7.10 $(a) - (f)$. For the smooth registration problem (Example 3), one can observe that firstly all three methods work fine in producing an acceptable registration and secondly the registered result by the new model $\mathcal{R}^{\text{NewCv}}$ is the best from both the visual effect and the value of $\widetilde{\varepsilon}_3$.

However, for the non-smooth registration problem (Example 4), one can clearly see that $\mathcal{R}^{\text{FMcurv}}$ and $\mathcal{R}^{\text{HWcurv}}$ failed to deliver a good registration (note other models from [104] cannot

[1] http://www.math.mu-luebeck.de/safir/

register this hard example either), from Figure 7.9 $(d)-(f)$ and Figure 7.10 $(d)-(f)$, while our new model $\mathcal{R}^{\text{NewCv}}$ evidently produced visually pleasing results. The main reason is that the exact deformation field should have a non-smooth shift for the left book to the top; c.f. Figure 7.10 (f). Precisely, this field is piecewise constant and substantially discontinuous at regions close to the interface of the books. Consequently, $\mathcal{R}^{\text{FMcurv}}$ and $\mathcal{R}^{\text{HWcurv}}$ must fail because they smooth the field at those regions; see over smoothing results of the field in Figure 7.10 $(d)-(e)$.

Both examples confirm that our new model $\mathcal{R}^{\text{NewCv}}$ is better and more robust than $\mathcal{R}^{\text{HWcurv}}$ and $\mathcal{R}^{\text{FMcurv}}$ [47, 48, 79, 78, 73, 75, 74] which are in turn better than a class of other registration models.

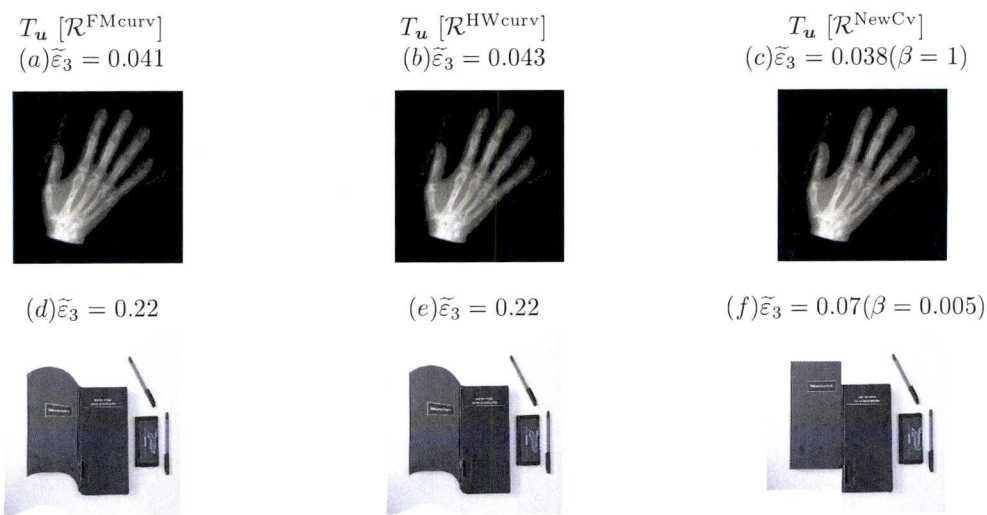


Figure 7.9: Registered images for Example $3-4$ shown in Figure 7.8 $(a)-(d)$. Left to right: results by (a) $\mathcal{R}^{\text{FMcurv}}$, (b) $\mathcal{R}^{\text{HWcurv}}$, and (c) $\mathcal{R}^{\text{NewCv}}$. Top to bottom: results from Example 3 (the smooth registration problem) and Example 4 (the non-smooth registration problem). Recall that $\widetilde{\varepsilon}_3$ means the relative reduction of the dissimilarity defined in Algorithm 7.4.2.

7.5.2 Tests of our new FAS-NMG algorithm

In the previous section we have used the LFA to inform our theoretical choice of suitable smoothers for our new FAS-NMG Algorithm 7.4.2. Here by experiments, we hope to first verify the reliability of this choice and then to further test the convergence issues of it with regard to parameters α, β in the model and the mesh parameter h.

1) Comparison of smoothers and $h-$independent convergence tests

We shall re-solve the same Examples $3-4$ as above using an increasing sequence of resolutions (or a decreasing mesh parameter h) and show the results in Table 7.3. Algorithm 7.4.2 is run using 3 separate smoothers (1 by Method 4 - SFP, 2 by Mehtod 5 - PDFP I and 2^* by a modified Method 5 - PDFP II respectively). In each case the algorithm is stopped when the mean of

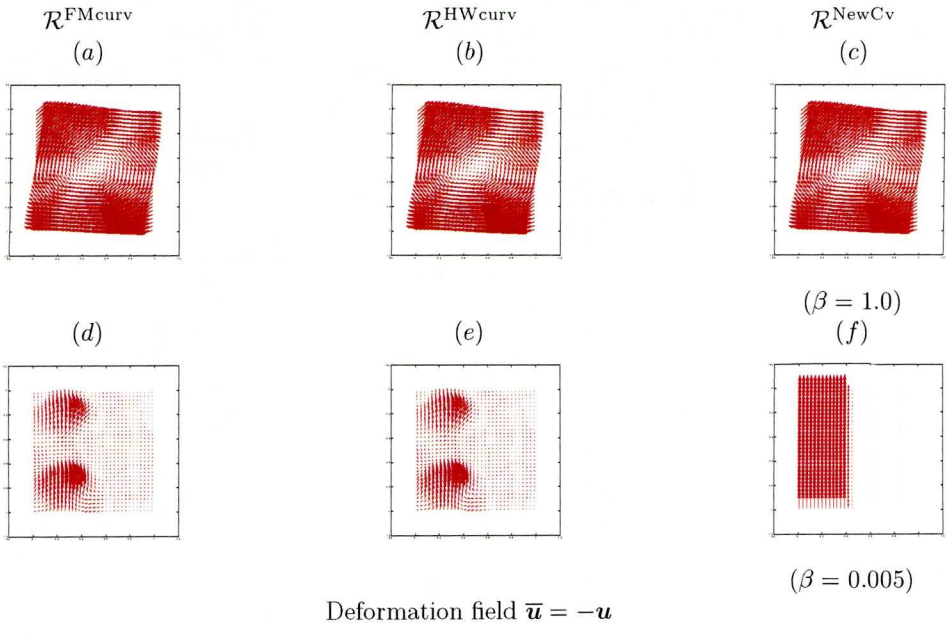


Figure 7.10: Recovered deformation fields for Example 3 – 4 shown in Figure 7.8 $(a) - (d)$. Left to right: results by (a) $\mathcal{R}^{\text{FMcurv}}$, (b) $\mathcal{R}^{\text{HWcurv}}$, and (c) $\mathcal{R}^{\text{NewCv}}$. Top to bottom: results from Example 3 (the smooth registration problem) and Example 4 (the non-smooth registration problem).

the relative residual below 10^{-6} with 'M' the recorded number multigrid cycles required. Then to get an measure of speed without using the machine-dependent CPUs, we work out the work units (WUs) for each case. We also use the relative reduction of dissimilarity $\widetilde{\varepsilon}_3$ to indicate the quality of registration obtained at cycle 'M'.

	MG with Smoother 1 (**SFP**) $\nu_1/\nu_2/PCGSiter$/M/D/WUs	MG with Smoother 2 (**PDFP I**) $\nu_1/\nu_2/PCGSiter$/M/D/WUs	MG with Smoother 2* (**PDFP II**) $\nu_1/\nu_2/PCGSiter$/M/D/WUs
Example 3	$\alpha = 1/10000, \gamma = 1/\sqrt{\beta}$		
$h = 1/128$	10/10/10/18/0.0258/480	10/10/10/6/0.0264/160	10/10/10/5/0.0258/133
$h = 1/256$	10/10/10/ * / * /*	10/10/10/7/0.0388/187	10/10/10/6/0.0386/160
$h = 1/512$	10/10/10/ * / * /*	10/10/10/7/0.0379/187	10/10/10/6/0.0379/160
$h = 1/1024$	10/10/10/ * / * /*	10/10/10/8/0.0412/213	10/10/10/7/0.0398/187
Example 4	$\alpha = 0.75/10000$		
$h = 1/128$	10/10/15/ * / * /*	10/10/15/11/0.0713/293	10/10/15/8/0.0698/213
$h = 1/256$	10/10/15/ * / * /*	10/10/15/12/0.0739/320	10/10/15/9/0.0701/240
$h = 1/512$	10/10/15/ * / * /*	10/10/15/12/0.0761/320	10/10/15/10/0.0712/267
$h = 1/1024$	10/10/15/ * / * /*	10/10/15/13/0.0793/347	10/10/15/10/0.0753/267

Table 7.3: Registration results of Algorithms 7.4.2 with the proposed smoothers for processing Examples 3 – 4 shown respectively in Figure 7.8 $(a) - (d)$. The letters 'M', 'D', and 'WUs' mean the number of multigrid cycles, the relative reduction of dissimilarity ($\widetilde{\varepsilon}_3$), the work units, respectively. '$*$' indicates failure in dropping the mean of the relative residual to 10^{-6} within 20 MG-cycles. Recall that γ is the SFP parameter.

Here we define a work unit used in measured computational work as the work of performing a smoother or relaxation step on the finest grid defined as follows:

1 WU = (cost of discretisating and constructing the linearised system per grid point
+ cost of PCGS updating per grid point)N (if N is the number of grid points)

For example, a work unit in performing one step of the PDFP I smoother can be estimated by

$$1 \text{ WU} = (150 + 123(PCGSiter))N$$

where each grid point in the linearised system (4×4) given in (7.30) is solved by the Gaussian elimination method, which has the cost of $\frac{(4)^3}{3} + \frac{(4)^2}{2} - \frac{5(4)}{6}$ additions and $\frac{(4)^3}{3} + (4)^2 - \frac{(4)}{3}$ multiplications. Therefore, the total costs of one V-cycle used L coarse grids can be estimated as follows:

$$\text{V-cycle cost} = (\nu_1 + \nu_2)(150 + 123(PCGSiter))N\sum_{k=0}^{L}(1/4)^k \ < \ \frac{4}{3}(\nu_1 + \nu_2) \text{ WUs.}$$

Here we have ignored the cost of interpolation and restriction procedures as well as the cost of residual correction procedure because they are relative small compared with that of smoothing procedures. Recall that ν_1, ν_2, and $PCGSiter$ denote respectively the number of pre- and post-smoothing and PCGS steps.

In the numerical results shown in Table 7.3, one can see six quantities: the numbers of pre- and post-smoothing and PCGS steps ν_1, ν_2; the multigrid cycles 'M'; the relative reduction of dissimilarity $D = \widetilde{\varepsilon}_3$ and WUs.

As expected from the LFA results in the last section, our numerical results confirm that Smoothers $2, 2^*$ (as PDFP I and II) are much better than Smoother 1 (SFP) for our FAS-NMG algorithm, because they not only lead to the convergence within a few MG cycles as expected of a multigrid technique, but also to the accurate results. The dissimilarities between the reference and registered images have been reduced more than 90% for both examples.

Overall, as LFA predicts, the above experimental results suggest that Smoother 2^* (PDFP II) would be preferred for practical applications. In other tests, we note the Smoother 1 can lead to the MG convergence for both registration problems when the number of pre- and post smoothing steps ν_1 and ν_2 are doubled.

2) $\alpha-$dependence tests

Next we assess how our MG algorithm is affected by varying α. To this end, the MG algorithm based on Smoother 2^* was tested on Example 3 (see and Figure 7.8 $(a) - (b)$) with the results shown in Table 7.4. Here the following parameters are used: $\beta = 1$, $\nu_1 = \nu_2 = PCGSiter = 10$, and $h = 1/256$ for all experiments and α is varied from $1/10000$ to $1/10$. For this example, large α is not needed as small ones give better results. However, the selection of suitable α is a separate but important issue because it is in general unknown a priori and it significantly effects on the qualities of registered images as well as the MG performance. In order to estimate a reasonable α automatically, we may adapt our MG algorithm and follow the 'cooling' process suggested in [33, 66, 65] which resembles the L-curve method in other inverse problems. Nevertheless, for the range of α tested in Table 7.4, our FAS-NMG remains efficient.

α	β	M	D
10^{-4}	1	6	0.0379
10^{-3}	1	7	0.1528
10^{-2}	1	7	0.3019
10^{-1}	1	15	0.4709

Table 7.4: Results for $\alpha-$dependence tests of Algorithms 7.4.2 with the PDFP II smoother for Example 3 shown in Figure 7.8$(a)-(b)$. The letters 'M' and 'D' mean the number of multigrid steps and the relative reduction of dissimilarity ($\widetilde{\varepsilon}_3$).

3) $\beta-$dependence tests

As is well known, the quantities of results and the performances of the MG techniques in solving the nonlinear system related to the TV regularisation technique are affected significantly by the values of β. As already discussed in Section 2, for registration purposes $\beta = 1$ is suitable for smooth registration problems because the diffusion coefficients ($\widetilde{D}_l$) are almost isotropic in all regions and then it leads to the smooth deformation fields. On the other hand $\beta << 1$ is appropriate for non-smooth registration problems because the diffusion coefficients are zero in regions representing large gradients of the fields and then it allows discontinuities at those regions. Here our aim is to see how our MG algorithm is affected by varying the values of β.

To this end, the MG algorithm based on Smoother 2* was tested on the non-smooth Example 4 as from Figure 7.8 $(c)-(d)$. Here the following parameters are taken: $\alpha = 0.75/10000$, $\nu_1 = \nu_2 = 10$, $PCGSiter = 15$, and $h = 1/256$ for all experiments and β is varied from 0.005 to 1. Table 7.5 shows that our MG algorithm converges in a few steps. Theoretically β should be selected to be as small as possible. However, in practice, small β is not necessary and not recommendable. As shown in our experiments, $\beta = 10^{-2}$ is enough to solve the non-smooth registration problem with the accurate results and with it our FAS-NMG algorithm has a fast convergence.

α	β	M	D
0.75×10^{-4}	5×10^{-3}	9	0.0701
0.75×10^{-4}	1×10^{-2}	8	0.0893
0.75×10^{-4}	1×10^{-1}	7	0.2324
0.75×10^{-4}	1×10^{-0}	6	0.4557

Table 7.5: Results for $\beta-$dependence tests of Algorithm 7.4.2 with Smoother 2* for Example 4 shown respectively in Figure 7.8 $(c)-(d)$. The letters 'M' and 'D' mean the number of multigrid steps and the relative reduction of dissimilarity ($\widetilde{\varepsilon}_3$).

7.6 Conclusions

The majority of deformable registration models in the variational framework use the gradient information (first order derivatives) in their regularisers. For problems requiring less smooth deformation fields, such models become ineffective and the curvature like information (second

order derivatives) used in regularisation can improve the registration results, as shown in the recent works of [47, 48, 79, 78, 73, 75, 74] where higher-order and essentially *linear PDEs* are solved.

Different from approximate curvature models of [47, 48, 79, 78, 73, 75, 74], the full curvature model proposed in this chapter does not make assumptions on the deformation fields. Consequently, our results shown in the previous section improve over previous approximate curvature models for both smooth and non-smooth registration problems in quality and robustness of image registration. Associated with the full curvature model is the apparent difficulty in developing a fast solution as the Euler-Lagrange equations of two coupled PDEs is highly nonlinear and of fourth order so standard unilevel methods are not appropriate. To end this, we proposed several iterative methods including the so-called *primal-dual fixed-point* (PDFP) method. As analysed its smoother properties by the LFA, the PDFP method was recommended to be a potential smoother in our FAS-NMG framework. Numerical experiments on synthetic and realistic images not only confirmed that the proposed curvature model is more robust in registration quality for a wide range of applications than the approximate curvature models of [47, 48, 79, 78, 73, 75, 74], but also that the FAS-NMG approach based on the proposed PDFP type smoother is fast and accurate in delivering visually-pleasing registration results.

Chapter 8

An Improved Monomodal Image Registration Model and Its Fast Algorithm

In previous chapters the image intensities of two given images are assumed to be comparable (i.e. in a monomodal registration scenario) and the so-called *sum of squared differences* is known as the proper choice to measure image similarities. In this chapter we relax this assumption and propose first a new variational model combining intensity and geometric transformations, as an alternative to using mutual information, and then its fast algorithm based on a multigrid strategy. This variational model allows one to solve a typical case of multimodal image registration where a given image has the similar features, but different intensity variations. Finally, we demonstrate the robustness of the proposed variational model and its numerical approach using clinical images.

8.1 Introduction

Under many real-world conditions, even intensity variations of two given images taken from the same object on the same scanner within the same protocol can be locally or globally different, e.g. clinical magnetic resonance (MR) images affected by the signal intensity inhomogeneity (bias field) due to imperfections in the radio frequency coils and object dependent interactions [97, 98, 101, 110, 144]. The sum of squared differences (SSD) without any pre-processing steps (e.g. the intensity normalisation or standardisation methods) is not suitable to measure image similarities as it reduces accuracy and efficiency of an expected registration; see e.g. a registration problem and its results shown in Figure 8.1 – 8.2. Mutual information (MI), on the other hand, is more appropriate and invariant to overall intensity scale differences. It is often adopted to deal with the lack of a model of intensity transformations. However, mutual information has a number of well-known drawbacks. Firstly, mutual information is known to be highly non-convex as well as nonlinear and has typically many local minima. Therefore non-linearity of the registration problem is enhanced by the usage of mutual information. Secondly the computations of mutual information and its first variation require approximations

of the joint density, which summarises the co-occurrence of events from the image intensities obtained from the given images. Such approximations are usually expensive and sensitive to some parameters, such as the width of the Parzen-window kernel and the set of local intensity samples. Finally, due to the mentioned difficulties, there is not a unique or even common implementation for estimating mutual information and its first variation; see more discussions in [37, 64, 95, 105, 107, 114, 119] and references therein.

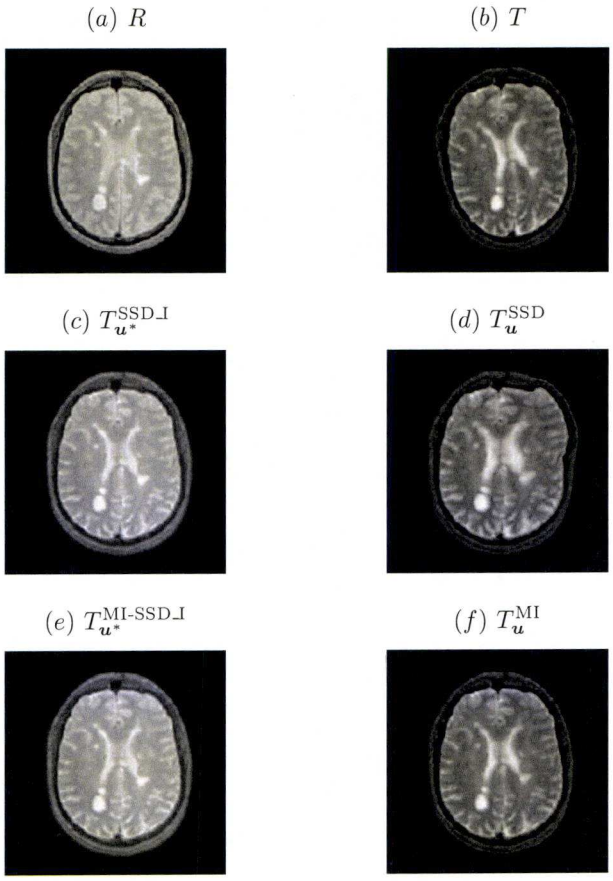


Figure 8.1: Numerical results by three similarity measures. Top row: a registration problem consisting a pair of MR image of a human head shown in (a) reference R and (b) template T. Middle row: two registered images (c) $T_{u^*}^{\text{SSD\_I}}$ by the proposed variational model (8.7) and (d) T_{u}^{SSD} by SSD. Bottom row: two registered images (e) $T_{u^*}^{\text{MI-SSD\_I}}$ by the proposed variational model (8.19) for the standardisation between R and T_{u}^{MI} and (f) T_{u}^{MI} by MI. Notice first that the model (8.7) accurately registers the images without any additional pre-processing steps. Second, the model (8.19) is effective in normalizing (post-processing) the intensity variations between the images.

Current registration models related to our work are found in [58, 113, 106, 111, 1]. In [58] the polynomial based intensity transformation is used in the elastic registration with an iterative scheme that alternates between estimating the coefficients of the polynomial and searching the non-parametric transformation minimising the energy functional using the demoms method [133]. These coefficients have the purpose to estimate the adequate intensity changes that

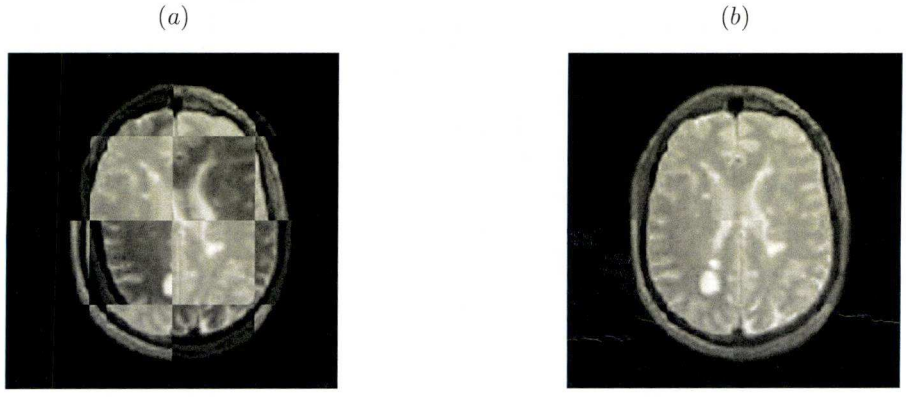


Figure 8.2: Composite views between the images before and after registration for the problem shown in Figure 8.1 $(a) - (b)$. (a) composite view between R and T before registration; (b) composite view between R and $T_{u^*}^{\mathrm{SSD\_I}}$ after registration based on our variational model (8.7). The intensity variations in (b) between the images are well-matched.

match the intensity values between the images. In [113], a locally linear intensity transformation with a smoothness constraint on the contrast/brightness parameters is used to model the elastic registration of multimodal images based on a locally affine transformation with a global smoothness constraint in a differential multiscale framework. In [106] the non-parametric intensity transformation is used in the elastic registration of monomodal images where the total variation (TV) energy is applied to constrain the intensity transformations. In [111], the registration of multimodal images is modelled by a low-order polynomial intensity transformation and a global affine transformation. In [1], the registration of multimodal images is modelled using a probabilistic formulation in a multiscale framework. The main aim is to simultaneously determine the local parameters of the geometric transformation using the B-spline models by [102] and the local coefficients of the polynomial intensity transformation that lead to successful registration. These parameters and coefficients are represented as Markov random fields giving the priori information about the homogeneity of the intensity and geometric changes.

In recent applications, a fast registration method becomes more and more important for high-resolution digital images. For a nonlinear system like (8.24) (see §8.3 later), the use of a nonlinear multigrid (NMG) method is natural and has been proven to be very successful in various image processing applications; see e.g. [6, 7, 13, 22, 33, 34, 53, 54, 61, 65, 76, 145] for either Euler-Lagrange systems of second- or fourth-order PDEs. Previous work on NMG techniques for deformable image registration using non-rigid deformations in [33, 34, 53, 54, 76, 145] considers different deformation models or different multigrid components. In [33, 34], the efficient NMG methods based on the typical fixed-point (FP) iteration method for overcoming the singular Neumann boundary problems of the discrete systems are presented respectively for the diffusion- and modified total variation-based image registration. In [53], a special treatment

for the singular systems due to the Neumann boundary conditions before and during the NMG method for TV-based image registration is introduced in solving the minimisation problem of the SSD functional. In [54], a full-multigrid (FMG) method based on the Newton-Gauss-Seidel smoother and an adaptive smoothing approach for the deformation field is developed in the context of diffusion image registration. In [76], a NMG method based on the discretised optimality conditions for elastic image registration is presented. In [145], a FMG method based on the FP type of smoothers is developed for diffusion image registration subject to Dirichlet boundary conditions. Although NMG techniques are used for other models of non-rigid image registration, to the best of our knowledge it has not been applied to solve the more challenging system of nonlinear PDEs like (8.24) for simultaneously determining the intensity and geometric transformations.

The rest of this chapter is organized as follows. A new variational image registration model combining the intensity and geometric transformations is proposed in §8.2 followed by its Euler-Lagrange equations with the corresponding primal-dual formulation in §8.3. §8.4–8.5 discuss the numerical implementation and the numerical solution for the primal-dual formulation, in particular a proposed multilevel approach based on an efficient NMG algorithm. The robustness of the proposed registration model and its numerical approach is illustrated using clinical data in §8.6. The last section is devoted for conclusions.

8.2 The proposed variational image registration model

A general framework of the registration problem of monomodal images can be re-formulated as follows: Given a *reference* R and a *template* T, we search simultaneously for a vector-valued non-parametric transformation φ defined by

$$\varphi(\boldsymbol{u})(\cdot) : \mathbb{R}^d \to \mathbb{R}^d, \qquad \varphi(\boldsymbol{u})(\mathbf{x}) : \mathbf{x} \mapsto \mathbf{x} + \boldsymbol{u}(\mathbf{x}) \tag{8.1}$$

that depends on an unknown *deformation* or *displacement field*

$$\boldsymbol{u} : \mathbb{R}^d \to \mathbb{R}^d, \qquad \boldsymbol{u} : \mathbf{x} \mapsto \boldsymbol{u}(\mathbf{x}) = (u_1(\mathbf{x}), u_2(\mathbf{x}), \ldots, u_d(\mathbf{x}))^\top. \tag{8.2}$$

and an intensity transformation f such that the transformed template

$$f(T \circ \varphi(\boldsymbol{u}(\mathbf{x}))) = f(T(\mathbf{x} + \boldsymbol{u}(\mathbf{x}))) = f(T_{\boldsymbol{u}}(\mathbf{x}))$$

becomes similar to the reference R in a geometric sense, i.e.

$$R(\mathbf{x}) = f(T_{\boldsymbol{u}}(\mathbf{x})) + \eta(\mathbf{x}). \tag{8.3}$$

Here $\eta(\mathbf{x})$ is random and uncorrelated noise. Recall that the given images R and T are modelled as the continuous functions mapping from an image domain $\Omega \subset \mathbb{R}^d$ into $V \subset \mathbb{R}_0^+$ and each component u_d of $\boldsymbol{u}$ is the function of the spatial position $\mathbf{x} = (x_1, x_2 \ldots, x_d)^\top \in \Omega$. Without loss of generality we assume that the registration problem is described in the two-dimensional case ($d = 2$) throughout this chapter, but it is readily extendable to the three-dimensional case

($d = 3$). We also assume further that $\Omega = [0,1]^2 \subset \mathbb{R}^2$ and $V = [0,1]$ for 2D gray intensity images.

If the intensity variations of R and T are comparable, the intensity transformation f can be represented by the identity function and we search only the deformation field $\boldsymbol{u}$. However, there are cases in real-life applications, as those found in MR or other medical imaging, where inhomogeneity of image intensities and noise are present in both images or one of them, and thus these cases require a more complex intensity function f than those of parametric intensity transformations like polynomial ones. To design a general-purpose registration model for these cases, let us model f to be non-parametric and assume that it depends independently on each position $\mathbf{x} \in \Omega$ with the following intensity relationships

$$\text{Intensity model I: } f(T_{\boldsymbol{u}}(\mathbf{x})) = T_{\boldsymbol{u}}(\mathbf{x}) + c(\mathbf{x}) \text{ (additive intensity correction model)} \tag{8.4}$$

$$\text{Intensity model II: } f(T_{\boldsymbol{u}}(\mathbf{x})) = c(\mathbf{x})T_{\boldsymbol{u}}(\mathbf{x}) \text{ (multiplicative intensity correction model)} \tag{8.5}$$

where $c : \Omega \to \mathbb{R}$ is an unknown non-parametric intensity correction. Note that the intensity model II in (8.5) is exactly the same as introduced by [106].

As usual for a non-parametric and non-rigid registration model, it requires first a suitable similarity functional to measure disparities between the given images and second a regularisation technique to rule out unwanted, irregular, and/or nonunique solutions. Since we search simultaneously for $\boldsymbol{u}$ and c, the registration problem can be posed as a minimisation problem as follows:

$$\min_{\boldsymbol{u},c}\{\mathcal{J}_{\alpha_1,\alpha_2}(\boldsymbol{u},c) = \mathcal{D}(\boldsymbol{u},c) + \alpha_1\mathcal{R}_1(\boldsymbol{u}) + \alpha_2\mathcal{R}_2(c)\} \tag{8.6}$$

where $\alpha_1, \alpha_2 > 0$ are the regularisation parameters.

For the choice of the similarity functional $\mathcal{D}$, it is enough to modify the SSD functional from

$$\mathcal{D}^{\text{SSD}}(\boldsymbol{u},c) = \tfrac{1}{2}\int_\Omega \left(T(\mathbf{x}+\boldsymbol{u}(\mathbf{x})) - R(\mathbf{x})\right)^2 d\mathbf{x},$$

to the intensity models I and II in (8.4) and (8.5) as follows:

$$\mathcal{D}_{\text{I}}^{\text{SSD}}(\boldsymbol{u},c) = \tfrac{1}{2}\int_\Omega \left(T(\mathbf{x}+\boldsymbol{u}(\mathbf{x})) + c(\mathbf{x}) - R(\mathbf{x})\right)^2 d\mathbf{x}, \tag{8.7}$$

$$\mathcal{D}_{\text{II}}^{\text{SSD}}(\boldsymbol{u},c) = \tfrac{1}{2}\int_\Omega \left(c(\mathbf{x})\,T(\mathbf{x}+\boldsymbol{u}(\mathbf{x})) - R(\mathbf{x})\right)^2 d\mathbf{x}. \tag{8.8}$$

We select either $\mathcal{D} = \mathcal{D}_{\text{I}}^{\text{SSD}}$ or $\mathcal{D} = \mathcal{D}_{\text{II}}^{\text{SSD}}$ for the variational formulation (8.6).

For the choice of $\mathcal{R}_1$, we adopt here the full curvature regularisation given by

$$\mathcal{R}_1(\boldsymbol{u}) = \mathcal{R}^{\text{NewCv}}(\boldsymbol{u}) = \sum_{l=1}^{2}\int_\Omega \Phi(\kappa(u_l))d\mathbf{x}, \tag{8.9}$$

where $\Phi(s) = \frac{1}{2}s^2$ using the mean curvature

$$\kappa(u_l) = \nabla\cdot\frac{\nabla u_l}{|\nabla u_l|_{\beta_1}} = \frac{(\beta_1+u_{l_{x_1}}^2)u_{l_{x_1x_1}} - 2u_{l_{x_1}}u_{l_{x_2}}u_{l_{x_1x_2}} + (\beta_1+u_{l_{x_2}}^2)u_{l_{x_2x_2}}}{(\beta_1+u_{l_{x_1}}^2+u_{l_{x_2}}^2)^{3/2}},\ \beta_1 > 0, \tag{8.10}$$

in our variational framework because (i) it does not require an affine pre-registration step and (ii) it is more flexible for both smooth and non-smooth registration problems than common

regularisers choices such as

$$\mathcal{R}^{\text{elas}}(\boldsymbol{u}) = \int_\Omega ((\mu/4)\sum_{l,m=1}^{2}(\partial_{x_l}u_m + \partial_{x_m}u_l)^2 + (\lambda/2)(\nabla\cdot\boldsymbol{u})^2)d\mathbf{x}, \tag{8.11}$$

$$\mathcal{R}^{\text{diff}}(\boldsymbol{u}) = \frac{1}{2}\sum_{l=1}^{2}\int_\Omega |\nabla u_l|^2\, d\mathbf{x}, \tag{8.12}$$

$$\mathcal{R}^{\text{FMcurv}}(\boldsymbol{u}) = \frac{1}{2}\sum_{l=1}^{2}\int_\Omega (\Delta u_l)^2 d\mathbf{x}, \tag{8.13}$$

$$\mathcal{R}^{\text{HWcurv}}(\boldsymbol{u}) = \frac{1}{2}\sum_{l=1}^{2}\int_\Omega (\Delta u_l)^2 - 2(u_{l_{x_1x_1}}u_{l_{x_2x_2}} - u^2_{l_{x_1x_2}})d\mathbf{x}, \tag{8.14}$$

and

$$\mathcal{R}^{\beta\text{TV}}(\boldsymbol{u}) = \sum_{l=1}^{2}\int_\Omega |\nabla u_l|_{\beta_1}\, d\mathbf{x}. \tag{8.15}$$

These regularisation techniques are known to be suitable for either smooth or non-smooth registration problems. Note, in [106] only $\mathcal{R}^{\text{elas}}$ was considered.

For the choice of $\mathcal{R}_2$, instead of selecting the first-order variational models

$$\mathcal{TV}(c) = \int_\Omega |\nabla c|_{\beta_2} d\mathbf{x} = \int_\Omega \sqrt{c_{x_1}^2 + c_{x_2}^2 + \beta_2} d\mathbf{x},\ \beta_2 > 0 \tag{8.16}$$

and

$$\mathcal{R}^{L_2}(c) = \int_\Omega |\nabla c|^2 d\mathbf{x} = \int_\Omega (c_{x_1}^2 + c_{x_2}^2) d\mathbf{x} \tag{8.17}$$

as used and discussed by [106] we propose the second-order variational model based on the mean curvature $\kappa(c) = \nabla\cdot\frac{\nabla c}{|\nabla c|_{\beta_2}}$ as follows:

$$\mathcal{K}(c) = \int_\Omega \Psi(\kappa(c))d\mathbf{x}, \tag{8.18}$$

where $\Psi(t) = \frac{1}{2}t^2$. The main reasons for this choice $\mathcal{R}_2(c) = \mathcal{K}(c)$ are as follows:

(1) $\mathcal{K}(c(\mathbf{x}) = a_1x_1 + a_2x_2 + a_3) = 0$ for $\mathbf{a} = (a_1, a_2, a_3)^\top \in \mathbb{R}^3$, i.e the non-trivial kernel of $\mathcal{K}$ consists only of the linear transformations, and consequently this energy is invariant under globally and locally linear intensity corrections. Compared with those of (8.16) and (8.17) $\mathcal{TV}(c(\mathbf{x}) = a_1x_1 + a_2x_2 + a_3) = 0$ or $\mathcal{R}^{L_2}(c(\mathbf{x}) = a_1x_1 + a_2x_2 + a_3) = 0$ if and only if $\mathbf{a} = \mathbf{0}$. This means that both $\mathcal{TV}$ and $\mathcal{R}^{L_2}$ do not allow non-trivial linear intensity corrections; see Figure 8.5.

(2) $\mathcal{K}$ preserves discontinuities of c because the diffusion coefficients of the Euler-Lagrange equations resulting from (8.6) are zero in regions representing large gradient of c, i.e. $1/|\nabla c|_{\beta_2} \to 0$ and $\nabla c\cdot\nabla\Psi'(\kappa(c))/|\nabla c|^3_{\beta_2} \to 0$ when $|\nabla c|_{\beta_2} \to \infty$; see (8.21) and (8.24) in §8.3 later. As a result, the corrected images by $\mathcal{K}$ are not blurred, different from those by $\mathcal{R}^{L_2}$; see Figure 8.3.

The above theoretical remarks can be tested through a registration problem with its numerical results shown in Figure 8.3–8.5. Clearly $\mathcal{K}(c)$ is more suitable; as mentioned, in a previous work of using (8.8), $\mathcal{R}_1(\boldsymbol{u}) = \mathcal{R}^{\text{elas}}(\boldsymbol{u})$ and $\mathcal{R}_2(c) = \mathcal{TV}(c)$ were used in [106].

Finally we have some additional remarks:

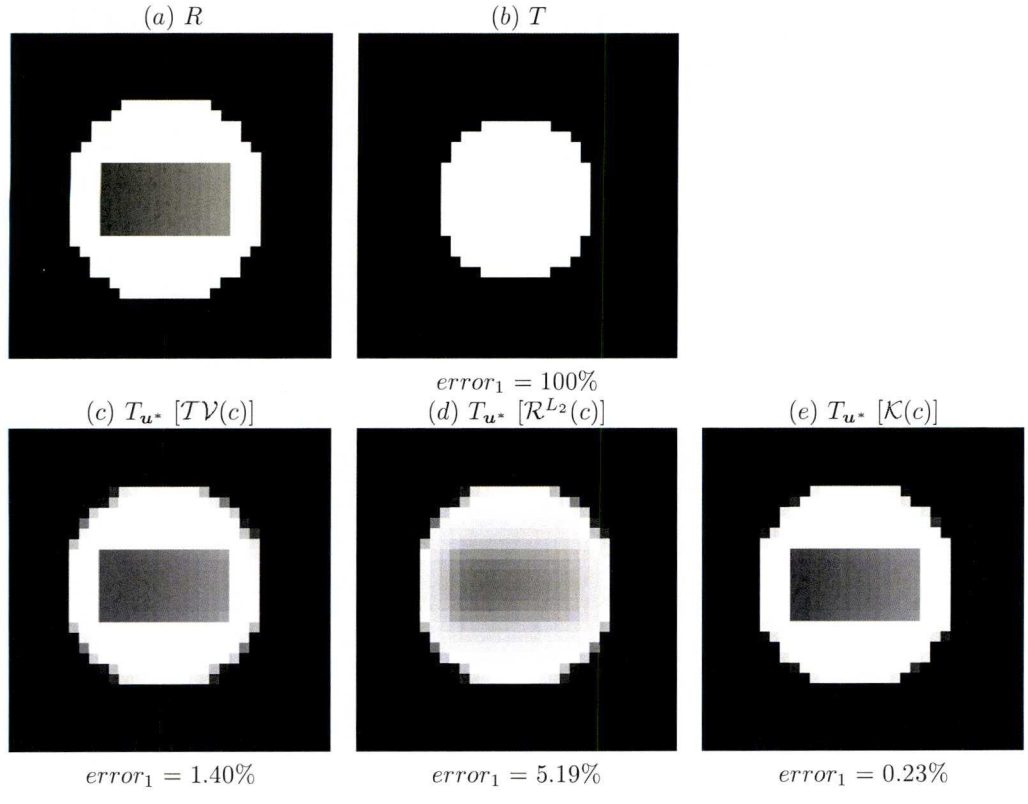


Figure 8.3: A numerical test with three regularisation techniques for c to show that our technique $\mathcal{K}(c)$ is better than $\mathcal{TV}(c)$ and $\mathcal{R}^{L_2}(c)$. Top row: a registration problem consisting a pair of two circles with a locally linear intensity variation shown in (a) reference R and (b) template T. Bottom row: three registered images $T_{\boldsymbol{u}^*}$ by $\mathcal{TV}(c)$, $\mathcal{R}^{L_2}(c)$, and $\mathcal{K}(c)$, respectively. Here $error_1$ denotes the percentage error.

Remark 8.2.1

(1) *If $\alpha_1 = 0$ and $\alpha_2 > 0$, the following minimisation problem*

$$\min_{c}\{\mathcal{J}_{\alpha_2}(\boldsymbol{u},c) = \mathcal{D}(\boldsymbol{u},c) + \alpha_2\mathcal{R}_2(c)\} \tag{8.19}$$

gives only the non-parametric intensity correction c for the normalisation or standardisation between the images, i.e. $T(\mathbf{x}) + c(\mathbf{x})$, $c(\mathbf{x})\,T(\mathbf{x}) \approx R(\mathbf{x})$ for fixing $\boldsymbol{u} = \boldsymbol{u}^{[0]}$, usually $\boldsymbol{u}^{[0]} = 0$; see e.g. the standardisation between R and $T^{MI}(\boldsymbol{u})$ using (8.19) in Figure 8.1 (e). That is to say, our reduced model (8.19) can be used for standardisation purpose, e.g. for post-processing MI results.

(2) *If $\alpha_1 > 0$ and $\alpha_2 = 0$, the following minimisation problem*

$$\min_{\boldsymbol{u}}\{\mathcal{J}_{\alpha_1}(\boldsymbol{u},c) = \mathcal{D}(\boldsymbol{u},c) + \alpha_1\mathcal{R}_1(\boldsymbol{u})\} \tag{8.20}$$

gives only the non-parametric deformation field $\boldsymbol{u}$ for the registration between the given

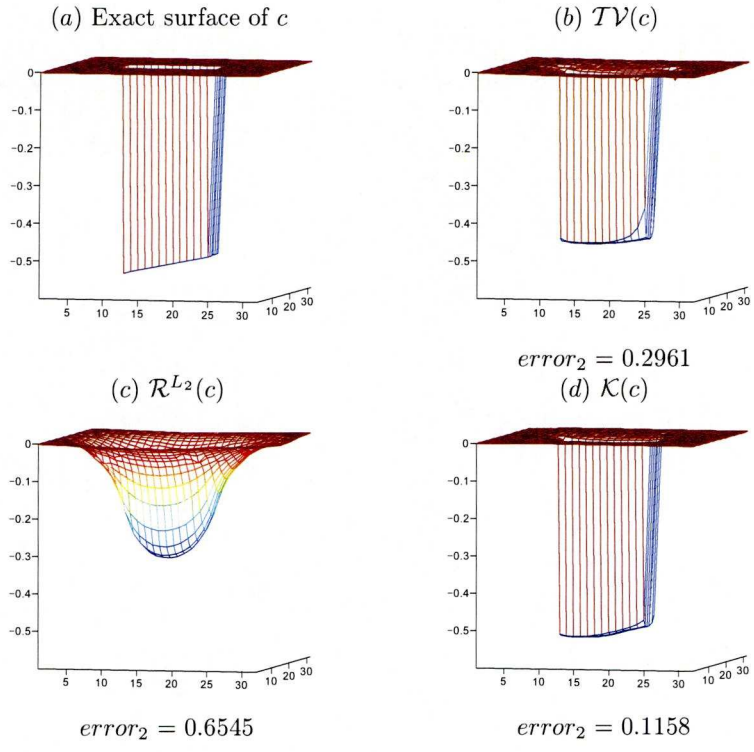


Figure 8.4: Surface plots of c for the registration problem in Figure 8.3 $(a)-(b)$. (a) the exact surface of c; $(b)-(d)$ the results by $\mathcal{TV}(c)$, $\mathcal{R}^{L_2}(c)$, and $\mathcal{K}(c)$, respectively. Here the $error_2$ denotes the 2-norm of the differences between the exact and approximate solutions.

images, i.e. $T(\mathbf{x}+\boldsymbol{u}(\mathbf{x}))+c(\mathbf{x}) \approx R(\mathbf{x})$ *for fixing* $c(\mathbf{x}) = 0$ *and* $c(\mathbf{x})T(\mathbf{x}+\boldsymbol{u}(\mathbf{x})) \approx R(\mathbf{x})$ *for fixing* $c(\mathbf{x}) = 1$.

(3) *The new variational model* (8.6) *can be adapted to solve problems related to optical flow computation or stereo disparity estimation, e.g. by introducing the energy funcational* $\mathcal{R}_2$ *in the variational formulation of optical flow computation.*

8.3 The Euler-Lagrange equations and its primal-dual formulation

Consider first the case $\mathcal{D} = \mathcal{D}_1^{\text{SSD}}$, $\mathcal{R}_1(\boldsymbol{u}) = \mathcal{R}^{\text{NewCv}}(\boldsymbol{u})$ and $\mathcal{R}_2(c) = \mathcal{K}(c)$. According to the calculus of variations, the resulting Euler-Lagrange equations of the proposed variational image

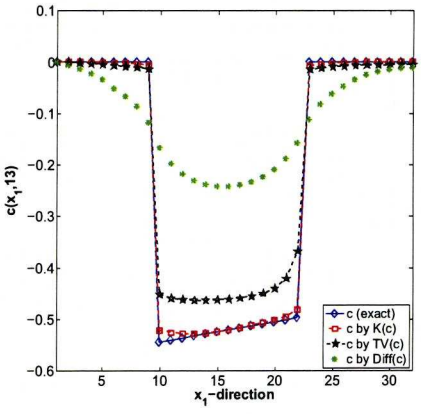


Figure 8.5: Plots of the 13th row of $c(x_1, x_2)$ in Figure 8.4 $(a)-(d)$ by $\mathcal{TV}(c)$, $\mathcal{R}^{L_2}(c)$, and $\mathcal{K}(c)$, respectively.

registration model (8.6) is given by

$$\begin{cases} \underbrace{(T_{\boldsymbol{u}}+c-R)\partial_{u_1}T_{\boldsymbol{u}}}_{f_1(\boldsymbol{u},c)} + \alpha_1\nabla\cdot\left(\frac{1}{|\nabla u_1|_{\beta_1}}\nabla\Phi'(\kappa(u_1)) - \frac{\nabla u_1\cdot\nabla\Phi'(\kappa(u_1))}{(|\nabla u_1|_{\beta_1})^3}\nabla u_1\right) = 0 \\ \underbrace{(T_{\boldsymbol{u}}+c-R)\partial_{u_2}T_{\boldsymbol{u}}}_{f_2(\boldsymbol{u},c)} + \alpha_1\nabla\cdot\left(\frac{1}{|\nabla u_2|_{\beta_1}}\nabla\Phi'(\kappa(u_2)) - \frac{\nabla u_2\cdot\nabla\Phi'(\kappa(u_2))}{(|\nabla u_2|_{\beta_1})^3}\nabla u_2\right) = 0 \\ \underbrace{(T_{\boldsymbol{u}}+c-R)}_{f_3(\boldsymbol{u},c)} + \alpha_2\nabla\cdot\left(\frac{1}{|\nabla c|_{\beta_2}}\nabla\Psi'(\kappa(c)) - \frac{\nabla c\cdot\nabla\Psi'(\kappa(c))}{(|\nabla c|_{\beta_2})^3}\nabla c\right) = 0 \end{cases} \tag{8.21}$$

subject to the boundary conditions

$$\langle\nabla u_l, \boldsymbol{n}\rangle_{\mathbb{R}^2} = 0,\ \langle\nabla\Phi'(\kappa(u_l)), \boldsymbol{n}\rangle_{\mathbb{R}^2} = 0 \text{ for } l = 1,\ 2 \text{ on } \partial\Omega \tag{8.22}$$

and

$$\langle\nabla c, \boldsymbol{n}\rangle_{\mathbb{R}^2} = 0,\ \langle\nabla\Psi'(\kappa(c)), \boldsymbol{n}\rangle_{\mathbb{R}^2} = 0 \text{ on } \partial\Omega. \tag{8.23}$$

Recall that the first and second terms in (8.21) are related to the first variations of $\mathcal{D}$ and $\mathcal{R}_{\bar{l}}$ $(\bar{l} = 1, 2)$, respectively.

As mentioned in §7.3.5, the primal-dual idea is suitable for solving a system of higher-order nonlinear PDEs like (8.21). The main idea is to reduce the order and nonlinearity of (8.21) using the new dual variables. Introducing additional unknown variables (dual variables)

$$v_1 = -\Phi'(\kappa(u_1)) = -\nabla\cdot\frac{\nabla u_1}{|\nabla u_1|_{\beta_1}},\ v_2 = -\Phi'(\kappa(u_2)) = -\nabla\cdot\frac{\nabla u_2}{|\nabla u_2|_{\beta_1}},\ v_3 = -\Psi'(\kappa(c)) = -\nabla\cdot\frac{\nabla c}{|\nabla c|_{\beta_2}}$$

leads (8.21) to the equivalent system of six second-order nonlinear PDEs given by

$$\begin{cases} -\nabla\cdot\frac{\nabla u_1}{|\nabla u_1|_{\beta_1}} - v_1 = g_1 \\ -\nabla\cdot\frac{\nabla u_2}{|\nabla u_2|_{\beta_1}} - v_2 = g_2 \\ -\nabla\cdot\frac{\nabla c}{|\nabla c|_{\beta_2}} - v_3 = g_3 \\ f_1(\boldsymbol{u},c) - \alpha_1\nabla\cdot\left(\frac{\nabla v_1}{|\nabla u_1|_{\beta_1}} + \frac{\nabla u_1\cdot(-\nabla v_1)}{|\nabla u_1|^3_{\beta_1}}\nabla u_1\right) = g_4 \\ f_2(\boldsymbol{u},c) - \alpha_1\nabla\cdot\left(\frac{\nabla v_2}{|\nabla u_2|_{\beta_1}} + \frac{\nabla u_2\cdot(-\nabla v_2)}{|\nabla u_2|^3_{\beta_1}}\nabla u_2\right) = g_5 \\ f_3(\boldsymbol{u},c) - \alpha_2\nabla\cdot\left(\frac{\nabla v_3}{|\nabla c|_{\beta_2}} + \frac{\nabla c\cdot(-\nabla v_3)}{|\nabla c|^3_{\beta_2}}\nabla c\right) = g_6 \end{cases} \tag{8.24}$$

subject to the boundary conditions transferred into

$$\langle\nabla u_1,\boldsymbol{n}\rangle_{\mathbb{R}^2} = \langle\nabla u_2,\boldsymbol{n}\rangle_{\mathbb{R}^2} = \langle\nabla c,\boldsymbol{n}\rangle_{\mathbb{R}^2} = \langle\nabla v_m,\boldsymbol{n}\rangle = 0 \text{ for } m = 1,2,3$$

where $\boldsymbol{g} = (g_1,g_2,g_3,g_4,g_5,g_6)^\top = \boldsymbol{0}$ on the finest grid for the MG setting in the coming section.

For the case $\mathcal{D} = \mathcal{D}_{\mathrm{II}}^{\mathrm{SSD}}$ only $f_l(\boldsymbol{u},c)$ (l = 1, 2, 3) in (8.21) and (8.24) need modifying and they are substituted respectively by

$$f_1(\boldsymbol{u},c) = c(cT'_{\boldsymbol{u}} - R)\partial_{u_1}T'_{\boldsymbol{u}}, \tag{8.25}$$

$$f_2(\boldsymbol{u},c) = c(cT'_{\boldsymbol{u}} - R)\partial_{u_2}T'_{\boldsymbol{u}}, \tag{8.26}$$

and

$$f_3(\boldsymbol{u},c) = (cT'_{\boldsymbol{u}} - R)T'_{\boldsymbol{u}}. \tag{8.27}$$

Refer to [106]. In this work $\mathcal{D} = \mathcal{D}_{\mathrm{I}}^{\mathrm{SSD}}$ is adopted in our numerical implementations because it is new for the variational model (8.6) and leads to a simple and efficient numerical scheme.

Here we have the following remarks:

Remark 8.3.1

(1) *If* $\mathcal{D} = \mathcal{D}_{II}^{SSD}$, $\mathcal{R}_1(c) = \mathcal{R}^{elas}(\boldsymbol{u})$, *and* $\mathcal{R}_2(c) = \mathcal{TV}(c)$, *the resulting Euler-Lagrange equations become*

$$\begin{cases} f_1(\boldsymbol{u},c) - \alpha_1\left((\lambda+2\mu)\partial_{x_1x_1}u_1 + \mu\partial_{x_2x_2}u_1 + (\lambda+\mu)\partial_{x_1x_2}u_2\right) = 0 \\ f_2(\boldsymbol{u},c) - \alpha_1\left((\lambda+\mu)\partial_{x_1x_2}u_1 + \mu\partial_{x_1x_1}u_2 + (\lambda+2\mu)\partial_{x_2x_2}u_2\right) = 0 \\ f_3(\boldsymbol{u},c) - \alpha_2\nabla\cdot\frac{\nabla c}{|\nabla c|_{\beta_2}} = 0 \end{cases} \tag{8.28}$$

subject to the boundary conditions

$$\left\langle\mu\left(\nabla\boldsymbol{u} + (\nabla\boldsymbol{u})^\top\right) + \lambda diag(\nabla\cdot\boldsymbol{u}),\boldsymbol{n}\right\rangle_{\mathbb{R}^2} = 0 \text{ and } \langle\nabla c,\boldsymbol{n}\rangle_{\mathbb{R}^2} = 0 \text{ on } \partial\Omega. \tag{8.29}$$

This is the PDE-based model used in [106] for combining homogenisation and registration.

(2) *If* $\mathcal{D} = \mathcal{D}_I^{SSD}$ *and* $\mathcal{R}_2(c) = \mathcal{TV}(c)$, *the last equation in* (8.21) *and the boundary conditions in* (8.23) *are replaced respectively by*

$$f_3(\boldsymbol{u},c) - \alpha_2\nabla\cdot\frac{\nabla c}{|\nabla c|_{\beta_2}} = 0 \tag{8.30}$$

and $\langle \nabla c, \boldsymbol{n} \rangle_{\mathbb{R}^2} = 0$ *on* $\partial\Omega$. *The resulting primal-dual formulation is then given by*

$$\begin{cases} -\nabla \cdot \frac{\nabla u_1}{|\nabla u_1|_{\beta_1}} - v_1 = g_1 \\ -\nabla \cdot \frac{\nabla u_2}{|\nabla u_2|_{\beta_1}} - v_2 = g_2 \\ f_1(\boldsymbol{u}, c) - \alpha_1 \nabla \cdot \left(\frac{\nabla v_1}{|\nabla u_1|_{\beta_1}} + \frac{\nabla u_1 \cdot (-\nabla v_1)}{|\nabla u_1|^3_{\beta_1}} \nabla u_1 \right) = g_3 \\ f_2(\boldsymbol{u}, c) - \alpha_1 \nabla \cdot \left(\frac{\nabla v_2}{|\nabla u_2|_{\beta_1}} + \frac{\nabla u_2 \cdot (-\nabla v_2)}{|\nabla u_2|^3_{\beta_1}} \nabla u_2 \right) = g_4 \\ f_3(\boldsymbol{u}, c) - \alpha_2 \nabla \cdot \frac{\nabla c}{|\nabla c|_{\beta_2}} = g_5 \end{cases} \tag{8.31}$$

(3) *If* $\mathcal{D} = \mathcal{D}_I^{SSD}$ *and* $\mathcal{R}_2(c) = \mathcal{R}^{L_2}(c)$, *the last equation in* (8.21) *and the boundary conditions in* (8.23) *become*

$$f_3(\boldsymbol{u}, c) - \alpha_2 \Delta c = 0 \tag{8.32}$$

and $\langle \nabla c, \boldsymbol{n} \rangle_{\mathbb{R}^2} = 0$ *on* $\partial\Omega$, *respectively. Similarly, the resulting primal-dual formulation is given by replacing the last equation in* (8.31) *with* $f_3(\boldsymbol{u}, c) - \alpha_2 \Delta c = g_5$.

8.4 Finite difference discretisation

For simplicity, let $(z_{\widehat{l}}^h)_{i,j} = z_{\widehat{l}}^h(x_{1_i}, x_{2_j})$ denote the grid functions for $\widehat{l} = 1, \ldots, 6$ where

$$\boldsymbol{z} = (z_1, z_2, z_3, z_4, z_5, z_6)^\top = (u_1, u_2, c, v_1, v_2, v_3)^\top \tag{8.33}$$

and let

$$\Omega_h = \{\mathbf{x} \in \Omega | \mathbf{x} = (x_{1_i}, x_{2_j})^\top = ((2i-1)h/2, (2j-1)h/2),\ 1 \le i, j \le n\} \tag{8.34}$$

be the discrete domain consisting of $N = n^2$ cells of size $h \times h$ with the grid mesh $h = 1/n$. The cell-centered finite difference approximations are used with the divergence terms $\nabla \cdot \boldsymbol{V}$ for any vector $\boldsymbol{V} = (V_1, V_2)$ in (8.24) at a grid point (i, j) as follows:

$$\left(\frac{\partial V_1}{\partial x_1}\right)_{i,j} + \left(\frac{\partial V_2}{\partial x_2}\right)_{i,j} = \frac{(V_1)_{i+1,j} - (V_1)_{i,j}}{h} + \frac{(V_2)_{i,j+1} - (V_2)_{i,j}}{h}. \tag{8.35}$$

Therefore, we need to calculate V_1 at the grid points $(i+1, j)$ and (i, j) and V_2 at the grid points $(i, j+1)$ and (i, j). We list here the approximations used in our numerical implementations for estimating V_1 at the grid point (i, j) as the following (discretisation for V_1 at the grid point $(i+1, j)$ and V_2 at the grid points $(i, j+1)$ and (i, j) can be given similarly):

$$\begin{aligned} \kappa(z_{\widehat{l}}^h)_{i,j} &= \left(\nabla \cdot \frac{\nabla z_{\widehat{l}}^h}{|\nabla z_{\widehat{l}}^h|_{\beta_*}}\right)_{i,j} \\ \left(\nabla \cdot \frac{\nabla z_{\widehat{l}}^h}{|\nabla z_{\widehat{l}}^h|_{\beta_*}}\right)_{i,j} &= \left[\frac{\delta_{x_1}^-}{h} \left(\frac{\delta_{x_1}^+(z_{\widehat{l}}^h)_{i,j}/h}{\sqrt{\beta_* + (\delta_{x_1}^+(z_{\widehat{l}}^h)_{i,j}/h)^2 + (\delta_{x_2}^+(z_{\widehat{l}}^h)_{i,j}/h)^2}} \right) \right. \\ &\quad \left. + \frac{\delta_{x_2}^-}{h} \left(\frac{\delta_{x_2}^+(z_{\widehat{l}}^h)_{i,j}/h}{\sqrt{\beta_* + (\delta_{x_1}^+(z_{\widehat{l}}^h)_{i,j}/h)^2 + (\delta_{x_2}^+(z_{\widehat{l}}^h)_{i,j}/h)^2}} \right) \right], \\ &= (1/h^2)((\Sigma_{\widehat{l}}^h)_{i,j}(z_{\widehat{l}}^h)_{i,j} - (\overline{\Sigma}_{\widehat{l}}^h)_{i,j}(z_{\widehat{l}}^h)_{i,j}), \\ (\Sigma_{\widehat{l}}^h)_{i,j} &= 2D_{\widehat{l}3}(z_{\widehat{l}}^h)_{i,j} + D_{\widehat{l}1}(z_{\widehat{l}}^h)_{i-1,j} + D_{\widehat{l}2}(z_{\widehat{l}}^h)_{i,j-1}, \end{aligned}$$

$$
\begin{aligned}
(\overline{\Sigma}_{\widehat{l}}^{h})_{i,j}(z_{\widehat{l}}^{h})_{i,j} &= (D_{\widehat{l}3}(z_{\widehat{l}}^{h})_{i,j}((z_{\widehat{l}}^{h})_{i+1,j} + (z_{\widehat{l}}^{h})_{i,j+1}) \\
&\quad + D_{\widehat{l}1}(z_{\widehat{l}}^{h})_{i,j}(z_{\widehat{l}}^{h})_{i-1,j} + D_{\widehat{l}2}(z_{\widehat{l}}^{h})_{i,j}(z_{\widehat{l}}^{h})_{i,j-1})), \\
D_{\widehat{l}1}(z_{\widehat{l}}^{h})_{i,j} &= D_{\widehat{l}}(z_{\widehat{l}}^{h})_{i-1,j},\ D_{\widehat{l}2}(z_{\widehat{l}}^{h})_{i,j} = D_{\widehat{l}}(z_{\widehat{l}}^{h})_{i,j-1}, \\
D_{\widehat{l}3}(z_{\widehat{l}}^{h})_{i,j} &= D_{\widehat{l}}(z_{\widehat{l}}^{h})_{i+1,j} = D_{\widehat{l}}(z_{\widehat{l}}^{h})_{i,j+1} = D_{\widehat{l}}(z_{\widehat{l}}^{h})_{i,j}, \\
D_{\widehat{l}}(z_{\widehat{l}}^{h})_{i,j} &= |\nabla(z_{\widehat{l}}^{h})_{i,j}|_{\beta_*}, \\
|\nabla(z_{\widehat{l}}^{h})_{i,j}|_{\beta_*} &= \sqrt{\beta_* + (\delta_{x_1}^{+}(z_{\widehat{l}}^{h})_{i,j}/h)^2 + (\delta_{x_2}^{+}(z_{\widehat{l}}^{h})_{i,j}/h)^2},\ \beta_* = \beta_1 \text{ or } \beta_2, \\
(z_{\widehat{l}_{x_1}}^{h})_{i,j} &= \delta_{x_1}^{+}(z_{\widehat{l}}^{h})_{i,j}/h,\ (z_{\widehat{l}_{x_2}}^{h})_{i,j} = \delta_{x_2}^{+}(z_{\widehat{l}}^{h})_{i,j}/h, \\
\delta_{x_1}^{\pm}(z_{\widehat{l}}^{h})_{i,j} &= \pm((z_{\widehat{l}}^{h})_{i\pm1,j} - (z_{\widehat{l}}^{h})_{i,j}),\ \delta_{x_2}^{\pm}(z_{\widehat{l}}^{h})_{i,j} = \pm((z_{\widehat{l}}^{h})_{i,j\pm1} - (z_{\widehat{l}}^{h})_{i,j}), \\
I_{i,j}^{h^*} &= I^h(i + (u_1^h)_{i,j}, j + (u_2^h)_{i,j}), \\
f_1^h\left(u_1^h, u_2^h, c^h\right)_{i,j} &= (I_{i,j}^{h^*} + c_{i,j} - R_{i,j}^h)((I_{i+1,j}^{h^*} - I_{i-1,j}^{h^*})/(2h)), \\
f_2^h\left(u_1^h, u_2^h, c^h\right)_{i,j} &= (I_{i,j}^{h^*} + c_{i,j} - R_{i,j}^h)((I_{i,j+1}^{h^*} - I_{i,j-1}^{h^*})/(2h)), \\
f_3^h\left(u_1^h, u_2^h, c^h\right)_{i,j} &= (I_{i,j}^{h^*} + c_{i,j} - R_{i,j}^h).
\end{aligned}
$$

Note that the finite difference approximations for (8.24) need to be modified at grid points near the image boundary $\partial\Omega_h$ using the homogeneous Neumann boundary conditions approximated by one-side differences for boundary derivatives:

$$
(z_{\widehat{l}}^{h})_{i,1} = (z_{\widehat{l}}^{h})_{i,2},\ (z_{\widehat{l}}^{h})_{i,n} = (z_{\widehat{l}}^{h})_{i,n-1},\ (z_{\widehat{l}}^{h})_{1,j} = (z_{\widehat{l}}^{h})_{2,j},\ (z_{\widehat{l}}^{h})_{n,j} = (z_{\widehat{l}}^{h})_{n-1,j}. \tag{8.36}
$$

8.5 The numerical solution for the formulation (8.24)

To obtain a fast numerical solution of the new formulation (8.24) similar to the so-called *primal-dual fixed-point* (PDFP) method as described in §7.5 would be desirable; see also [6, 7, 13, 26, 27, 122, 136, 137] on FP schemes applied to other variational models involing TV operator.

8.5.1 A potential PDFP method

We now discuss a numerical scheme PDFP to solve the discrete version of (8.24). This is done in two steps:

I) The outer iteration step. Firstly, we introduce a new fixed-point or outer iteration to (8.24). This can be done as follows. Our scheme is *semi-implicit* in both regularisation and data terms. The semi-implicit scheme for the regularisation terms is iterated by freezing some coefficients in the similar ways with the Lagged-diffusivity method [26] or Quasi-Newton scheme [137, 136]. Starting with an initial guess $\boldsymbol{u}^{[0]}$ (e.g. $\boldsymbol{u}^{[0]} = 0$) leads to

$$
\mathbf{N}[\boldsymbol{z}^{[\nu]}]\boldsymbol{z}^{[\nu+1]} = \mathbf{G}[\boldsymbol{z}^{[\nu]}] \tag{8.37}
$$

where the typical Taylor's expansion for $f_l(u_1^{[\nu+1]}, u_2^{[\nu+1]}, c^{[\nu+1]})$ of type

$$\begin{aligned} f_l(u_1^{[\nu+1]}, u_2^{[\nu+1]}, c^{[\nu+1]}) &\approx f_l(c^{[\nu]}, u_1^{[\nu]}, u_2^{[\nu]}) + \partial_{u_1} f_l(u_1^{[\nu]}, u_2^{[\nu]}, c^{[\nu]})\delta u_1^{[\nu]} \\ &\quad + \partial_{u_2} f_l(u_1^{[\nu]}, u_2^{[\nu]}, c^{[\nu]})\delta u_2^{[\nu]} + \partial_c f_l(u_1^{[\nu]}, u_2^{[\nu]}, c^{[\nu]})\delta c^{[\nu]} \\ &= f_l(u_1^{[\nu]}, u_2^{[\nu]}, c^{[\nu]}) + \sigma_{l1}^{[\nu]}(u_1^{[\nu+1]} - u_1^{[\nu]}) + \sigma_{l2}^{[\nu]}(u_2^{[\nu+1]} - u_2^{[\nu]}) \\ &\quad + \sigma_{l3}^{[\nu]}(c^{[\nu+1]} - c^{[\nu]}) \end{aligned} \tag{8.38}$$

is used in the global linearisation scheme. Here

$$\begin{aligned} \sigma_{l1}^{[\nu]} &= \partial_{u_1} f_l(c^{[\nu]}, u_1^{[\nu]}, u_2^{[\nu]}) = (\partial_{u_l} T'_{\boldsymbol{u}^{[\nu]}})(\partial_{u_1} T'_{\boldsymbol{u}^{[\nu]}}) \\ &\quad + (T'_{\boldsymbol{u}^{[\nu]}} + c^{[\nu]} - R)(\partial_{u_1 u_l} T'_{\boldsymbol{u}^{[\nu]}}), \end{aligned} \tag{8.39}$$

$$\begin{aligned} \sigma_{l2}^{[\nu]} &= \partial_{u_2} f_l(, c^{[\nu]} u_1^{[\nu]}, u_2^{[\nu]}) = (\partial_{u_l} T'_{\boldsymbol{u}^{[\nu]}}))(\partial_{u_2} T'_{\boldsymbol{u}^{[\nu]}}) \\ &\quad + (T'_{\boldsymbol{u}^{[\nu]}} + c^{[\nu]} - R)(\partial_{u_2 u_l} T'_{\boldsymbol{u}^{[\nu]}}), \end{aligned} \tag{8.40}$$

$$\sigma_{l3}^{[\nu]} = \partial_c f_l(, c^{[\nu]} u_1^{[\nu]}, u_2^{[\nu]}) = (\partial_{u_l} T'_{\boldsymbol{u}^{[\nu]}}), \tag{8.41}$$

for $l = 1, 2$ and

$$\sigma_{31}^{[\nu]} = \partial_{u_1} f_3(c^{[\nu]}, u_1^{[\nu]}, u_2^{[\nu]}) = \partial_{u_1} T'_{\boldsymbol{u}^{[\nu]}}, \tag{8.42}$$

$$\sigma_{32}^{[\nu]} = \partial_{u_2} f_3(, c^{[\nu]} u_1^{[\nu]}, u_2^{[\nu]}) = \partial_{u_2} T'_{\boldsymbol{u}^{[\nu]}}, \tag{8.43}$$

$$\sigma_{33}^{[\nu]} = \partial_c f_3(, c^{[\nu]} u_1^{[\nu]}, u_2^{[\nu]}) = 1. \tag{8.44}$$

$$\mathbf{N}[\boldsymbol{z}^{[\nu]}] = \begin{bmatrix} -\mathcal{L}_1[u_1^{[\nu]}] & 0 & 0 & -1 & 0 & 0 \\ 0 & -\mathcal{L}_2[u_2^{[\nu]}] & 0 & 0 & -1 & 0 \\ 0 & 0 & -\mathcal{L}_3[c^{[\nu]}] & 0 & 0 & -1 \\ \sigma_{11}^{[\nu]} & \sigma_{12}^{[\nu]} & \sigma_{13}^{[\nu]} & -\alpha_1 \mathcal{L}_1[u_1^{[\nu]}] & 0 & 0 \\ \sigma_{21}^{[\nu]} & \sigma_{22}^{[\nu]} & \sigma_{23}^{[\nu]} & 0 & -\alpha_1 \mathcal{L}_2[u_2^{[\nu]}] & 0 \\ \sigma_{31}^{[\nu]} & \sigma_{32}^{[\nu]} & \sigma_{33}^{[\nu]} & 0 & 0 & -\alpha_2 \mathcal{L}_3[c^{[\nu]}] \end{bmatrix} \tag{8.45}$$

$$\begin{aligned} \boldsymbol{z}^{[\nu+1]} &= (z_1^{[\nu+1]}, z_2^{[\nu+1]}, z_3^{[\nu+1]}, z_4^{[\nu+1]}, z_5^{[\nu+1]}, z_6^{[\nu+1]})^\top, \\ &= (u_1^{[\nu+1]}, u_2^{[\nu+1]}, c^{[\nu+1]}, v_1^{[\nu+1]}, v_2^{[\nu+1]}, v_3^{[\nu+1]})^\top, \end{aligned} \tag{8.46}$$

$$\mathbf{G}[\boldsymbol{z}^{[\nu]}] = \boldsymbol{g} = (g_1, g_2, g_3, \widehat{g}_4^{[\nu]}, \widehat{g}_5^{[\nu]}, \widehat{g}_6^{[\nu]})^\top, \tag{8.47}$$

$$\widehat{g}_4^{[\nu]} = g_4 - f_1(u_1^{[\nu]}, u_2^{[\nu]}, c^{[\nu]}) + \sigma_{11}^{[\nu]} u_1^{[\nu]} + \sigma_{12}^{[\nu]} u_2^{[\nu]} + \sigma_{13}^{[\nu]} c^{[\nu]} + \alpha_1 \nabla \cdot (\frac{\nabla u_1^{[\nu]} \cdot (-\nabla v_{1)}^{[\nu]}}{|\nabla u_1^{[\nu]}|_\beta^3} \nabla u_1^{[\nu]}), \tag{8.48}$$

$$\widehat{g}_5^{[\nu]} = g_5 - f_2(u_1^{[\nu]}, u_2^{[\nu]}, c^{[\nu]}) + \sigma_{21}^{[\nu]} u_1^{[\nu]} + \sigma_{22}^{[\nu]} u_2^{[\nu]} + \sigma_{23}^{[\nu]} c^{[\nu]} + \alpha_1 \nabla \cdot (\frac{\nabla u_2^{[\nu]} \cdot (-\nabla v_{2)}^{[\nu]}}{|\nabla u_2^{[\nu]}|_\beta^3} \nabla u_2^{[\nu]}), \tag{8.49}$$

$$\widehat{g}_6^{[\nu]} = g_6 - f_3(u_1^{[\nu]}, u_2^{[\nu]}, c^{[\nu]}) + \sigma_{31}^{[\nu]} u_1^{[\nu]} + \sigma_{32}^{[\nu]} u_2^{[\nu]} + \sigma_{33}^{[\nu]} c^{[\nu]} + \alpha_2 \nabla \cdot (\frac{\nabla c^{[\nu]} \cdot (-\nabla v_{3)}^{[\nu]}}{|\nabla c^{[\nu]}|_\beta^3} \nabla c^{[\nu]}), \tag{8.50}$$

and

$$\mathcal{L}_m[z_m^{[\nu]}] z_{\widehat{l}}^{[\nu+1]} = \nabla \cdot (\overbrace{\frac{1}{|\nabla z_m^{[\nu]}|_{\beta_1}}}^{D_m(z_m^{[\nu]})} \nabla z_{\widehat{l}}^{[\nu+1]}) \ (m = 1, 2, 3 \text{ and } \widehat{l} = m \text{ or } \widehat{l} = m + 3). \tag{8.51}$$

II) The inner iteration step. After applying the finite difference approximations represented in §8.4 with (8.37), the PCGS relaxation method is used as the inner solver to solve inexactly the associate linear system. Here the kth PCGS step is given by

$$(\boldsymbol{z}^{[\nu+1]})_{i,j}^{[k+1]} = (\mathbf{N}[\boldsymbol{z}^{[\nu]}]_{i,j})^{-1}(\mathbf{G}[\boldsymbol{z}^{[\nu]}])_{i,j}^{[k+1/2]}, \tag{8.52}$$

where the symbol of the mesh parameter h is dropped for simplicity. We note that other choices of iterative techniques such as the line relaxation techniques or the preconditioned conjugate gradient method are optional. However, they are computationally more expensive than the PCGS relaxation method.

As discussed in §7.5, the PCGS relaxation method is not suitable to be a potential (multigrid) smoother for non-smooth problems. High values of the smoothing factors appear especially at the jumps of the coefficients $D_m(z_m^{[\nu]})_{i,j}$ compared with their neighborhood points. To avoid this situation, we introduce the so-called relaxation parameter $\omega \in (0,2)$, typically $\omega = 0.7$, and iterate the ω−PCGS steps at those odd points by

$$(\boldsymbol{z}^{[\nu+1]})_{i,j}^{[k+1]} = (1-\omega)\,(\boldsymbol{z}^{[\nu+1]})_{i,j}^{[k]} + \omega\underbrace{(\mathbf{N}[\boldsymbol{z}^{[\nu]}]_{i,j})^{-1}(\mathbf{G}[\boldsymbol{z}^{[\nu]}])_{i,j}^{[k+1/2]}}_{\text{original PCGS result}}, \tag{8.53}$$

with the following notation

$$\mathbf{N}[\boldsymbol{z}^{[\nu]}]_{i,j} = \frac{1}{h^2}\begin{bmatrix} (\Sigma_1^{[\nu]})_{i,j} & 0 & 0 & -1 & 0 & 0 \\ 0 & (\Sigma_2^{[\nu]})_{i,j} & 0 & 0 & -1 & 0 \\ 0 & 0 & (\Sigma_3^{[\nu]})_{i,j} & 0 & 0 & -1 \\ h^2\sigma_{11}^{[\nu]} & h^2\sigma_{12}^{[\nu]} & h^2\sigma_{13}^{[\nu]} & \alpha_1(\Sigma_1^{[\nu]})_{i,j} & 0 & 0 \\ h^2\sigma_{21}^{[\nu]} & h^2\sigma_{22}^{[\nu]} & h^2\sigma_{23}^{[\nu]} & 0 & \alpha_1(\Sigma_2^{[\nu]})_{i,j} & 0 \\ h^2\sigma_{31}^{[\nu]} & h^2\sigma_{32}^{[\nu]} & h^2\sigma_{33}^{[\nu]} & 0 & 0 & \alpha_2(\Sigma_3^{[\nu]})_{i,j} \end{bmatrix} \tag{8.54}$$

and

$$(\mathbf{G}[\boldsymbol{z}^{[\nu]}])_{i,j}^{[k+1/2]} = \begin{pmatrix} (g_1)_{i,j} + (1/h^2)(\overline{\Sigma}_1^{[\nu]})_{i,j}(u_1^{[\nu+1]})_{i,j}^{[k+1/2]} \\ (g_2)_{i,j} + (1/h^2)(\overline{\Sigma}_2^{[\nu]})_{i,j}(u_2^{[\nu+1]})_{i,j}^{[k+1/2]} \\ (g_3)_{i,j} + (1/h^2)(\overline{\Sigma}_3^{[\nu]})_{i,j}(c^{[\nu+1]})_{i,j}^{[k+1/2]} \\ (\widehat{g}_4)_{i,j}^{[\nu]} + (\alpha_1/h^2)(\overline{\Sigma}_1^{[\nu]})_{i,j}(v_1^{[\nu+1]})_{i,j}^{[k+1/2]} \\ (\widehat{g}_5)_{i,j}^{[\nu]} + (\alpha_1/h^2)(\overline{\Sigma}_2^{[\nu]})_{i,j}(v_2^{[\nu+1]})_{i,j}^{[k+1/2]} \\ (\widehat{g}_6)_{i,j}^{[\nu]} + (\alpha_2/h^2)(\overline{\Sigma}_3^{[\nu]})_{i,j}(v_3^{[\nu+1]})_{i,j}^{[k+1/2]} \end{pmatrix}. \tag{8.55}$$

Finally our proposed solver can be summarised as follows:

Algorithm 8.5.1 (Our Proposed Iterative Solver: PDFP)

Denote by

α regularisation parameter

ω relaxation parameter

$K > 0$ tolerance (typically $K = 0.5/\sqrt{\beta^*}$ where $\beta^* = \min\{\beta_1, \beta_2\}$)

$PCGSiter$ the maximum number of PCGS iterations

$[\overline{\boldsymbol{z}}^h] \leftarrow Solver\,(\overline{\boldsymbol{z}}^h, \boldsymbol{g}^h, R^h, T^h, \alpha, \omega, K, PCGSiter)$

- Use input parameters to compute $(\sigma_{lm})_{i,j}$, $(\mathbf{G}_h[\overline{\boldsymbol{z}}^h])_{i,j}$, and $(\mathbf{N}_h[\overline{\boldsymbol{z}}^h]_{i,j})^{-1}$ for $l, m = 1, 2, 3$ and $1 \leq i, j \leq n$
- Perform PCGS steps
  - for $k = 1 : PCGSiter$
    - for $i = 1 : n$
      - for $j = 1 : n$
        - if $D_m(\overline{z}^h_m)_{i,j} \geq K \cdot \min\{D_{m1}(\overline{z}_m)_{i,j}, D_{m2}(\overline{z}_m)_{i,j}, D_{m3}(\overline{z}_m)_{i,j}\}$ for $m = 1$, 2 or 3
          - Set $\omega = 0.7$

          else
          - Set $\omega = 1.0$

          end
        - Compute$(\widetilde{\boldsymbol{z}}^h)^{[k+1]}_{i,j}$ using (8.52)
        - $(\overline{\boldsymbol{z}})^{[k+1]}_{i,j} = (1-\omega)(\overline{\boldsymbol{z}})^{[k]}_{i,j} + \omega(\widetilde{\boldsymbol{z}}^h)^{[k+1]}_{i,j}$
      - end
    - end
  - end

8.5.2 A nonlinear multigrid algorithm

Below we apply the FAS-NMG method to solve (8.24), the coupled system of six second-order nonlinear PDEs given by

$$\mathcal{N}^h(\boldsymbol{z}^h) = \boldsymbol{g}^h, \text{ i.e. } \begin{cases} \mathcal{N}^h_1(\boldsymbol{z}^h) = g^h_1 \\ \vdots \\ \mathcal{N}^h_6(\boldsymbol{z}^h) = g^h_6 \end{cases} \tag{8.56}$$

in a similar way as presented in §7.4. In our FAS-NMG framework, the PDFP method represented in §8.5.1 is applied as the MG smoother and the standard coarsening is used for computing the coarse-grid domain Ω_H by doubling the grid size in each space direction, i.e. $h \rightarrow 2h = H$. For intergrid transfer operators between Ω_h and Ω_H, the averaging and bi-linear interpolation techniques are used for the restriction and interpolation operators denoted respectively by I^H_h and I^h_H. In order to compute the coarse-grid operator of $\mathcal{N}^h_l(\boldsymbol{z}^h)$ given by the left hand side of (8.24), the DCA method is employed. The pseudo-code implementation of our FAS-NMG method can be summarised in the following algorithm:

Algorithm 8.5.2 (FAS-NMG Algorithm)

Denote FAS-NMG parameters as follows:

ν_1 pre-smoothing steps on each level

ν_2 post-smoothing steps on each level

μ the number of multigrid cycles on each level (μ= 1 for V−cycling and μ= 2 for W−cycling)
[Here we present the V−cycle with μ= 1]

α regularisation parameter

ω relaxation parameter

$K > 0$ tolerance

$PCGSiter$ the maximum number of iterations using a smoother

$$\overline{\boldsymbol{z}}^h \leftarrow FASMG\left(\overline{\boldsymbol{z}}^h, \alpha, \overrightarrow{\varepsilon}\right)$$

- Select $\alpha, \overrightarrow{\varepsilon} = (\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4)$ and initial guess solutions $\widetilde{\boldsymbol{z}}^h_{initial}$ on the finest grid
- Set $K = 0$, $(\overline{\boldsymbol{z}}^h)^{[K]} = \widetilde{\boldsymbol{z}}^h_{initial}$, $\widetilde{\varepsilon}_2 = \varepsilon_2 + 1$, $\widetilde{\varepsilon}_3 = \varepsilon_3 + 1$, and $\widetilde{\varepsilon}_4 = \varepsilon_4 + 1$
- While ($K < \varepsilon_1$ AND $\widetilde{\varepsilon}_2 \geq \varepsilon_2$ AND $\widetilde{\varepsilon}_3 \geq \varepsilon_3$ AND $\widetilde{\varepsilon}_4 \geq \varepsilon_4$)
  - $(\overline{\boldsymbol{z}}^h)^{[K+1]} \leftarrow FASCYC((\overline{\boldsymbol{z}}^h)^{[K]}, \boldsymbol{g}^h, R^h, T^h, \nu_1, \nu_2, \alpha, \omega, K, PCGSiter)$
  - $\widetilde{\varepsilon}_2 = mean\{\|g^h_{\widehat{l}} - \mathcal{N}^h_{\widehat{l}}((\overline{\boldsymbol{z}}^h)^{[K+1]})\|_2 / \|g^h_{\widehat{l}} - \mathcal{N}^h_{\widehat{l}}(\widetilde{\boldsymbol{z}}^h_{initial})\|_2 \mid \widehat{l} = 1, \ldots, 6\}$
  - $\widetilde{\varepsilon}_3 = \overline{\mathcal{D}}^h(R^h, T^h_\star(\overline{\boldsymbol{u}}^h)^{[K+1]}) / \overline{\mathcal{D}}^h(R^h, T^h_\star(\overline{\boldsymbol{u}}^h)^{[0]})$,
    [Recall that $\overline{\mathcal{D}}^h\left(R^h, T^h_\star(\cdot)\right) \sim \frac{h^2}{2}\|R^h, T^h_\star(\cdot)\|^2_2$]
  - $\widetilde{\varepsilon}_4 = |\overline{\mathcal{D}}^h(R^h, T^h_\star(\overline{\boldsymbol{u}}^h)^{[K+1]}) - \overline{\mathcal{D}}^h(R^h, T^h_\star(\overline{\boldsymbol{u}}^h)^{[K]})|$
  - $K = K + 1$
- end

---

where

$$\left[\overline{\boldsymbol{z}}^h\right] \leftarrow FASCYC(\overline{\boldsymbol{z}}^h, \boldsymbol{g}^h, R^h, T^h, \nu_1, \nu_2, \alpha, \omega, K, PCGSiter)$$

- If $\Omega_h = coarset\ grid$ ($|\Omega_h| = 8 \times 8$), solve (8.24) using Algorithm 8.5.1 and then stop. Else continue with following step.
- Pre-smoothing:
  For $k = 1$ to ν_1, $\left[\overline{\boldsymbol{z}}^h\right] \leftarrow Solver(\overline{\boldsymbol{z}}^h, \boldsymbol{g}^h, R^h, T^h, \alpha, \omega, K, PCGSiter)$
- Restriction to the coarse grid:
  $\overline{z}^H_{\widehat{l}} \leftarrow I^H_h \overline{z}^h_{\widehat{l}}$ (for $\widehat{l} = 1, \ldots, 6$), $R^H \leftarrow I^H_h R^h$, $T^H \leftarrow I^H_h T^h$
- Set the initial solution for the coarse-grid problem:
  $\widetilde{z}^H_{\widehat{l}} \leftarrow \overline{z}^H_{\widehat{l}}$
- Compute the new right-hand side for the coarse-grid problem:
  $g^H_{\widehat{l}} \leftarrow I^H_h(g^h_{\widehat{l}} - \mathcal{N}^h_{\widehat{l}}(\overline{\boldsymbol{z}}^h)) + \mathcal{N}^H_{\widehat{l}}\left(\overline{\boldsymbol{z}}^H\right)$ (for $\widehat{l} = 1, \ldots, 6$)
- Implement the FAS-NMG method on the coarse-grid problem:
  For $k = 1$ to μ, $\left[\overline{\boldsymbol{z}}^H\right] \leftarrow FASCYC\left(\overline{\boldsymbol{z}}^H, \boldsymbol{g}^H, R^H, T^H, \nu_1, \nu_2, \alpha, \omega, K, PCGSiter\right)$
- Add the coarse-grid corrections:
  $\overline{z}^h_{\widehat{l}} \leftarrow \overline{z}^h_{\widehat{l}} + I^h_H\left(\overline{z}^H_{\widehat{l}} - \widetilde{z}^H_{\widehat{l}}\right)$, (for $\widehat{l} = 1, \ldots, 6$)
- Post-smoothing:
  For $k = 1$ to ν_2, $\left[\overline{\boldsymbol{z}}^h\right] \leftarrow Solver(\overline{\boldsymbol{z}}^h, \boldsymbol{g}^h, R^h, T^h, \alpha, \omega, K, PCGSiter)$

For practical applications our FAS-NMG approach is stopped if the maximum number of V− or W−cycles ε_1 is reached (usually $\varepsilon_1 = 20$), the mean of the relative residuals obtained from the Euler-Lagrange equations (8.24) is smaller than a small number $\varepsilon_2 > 0$ (typically $\varepsilon_2 = 10^{-4}$), the relative reduction of the dissimilarity is smaller than some $\varepsilon_3 > 0$ (we usually assign $\varepsilon_3 = 0.10$ meaning that the relative reduction of the dissimilarity would decrease about 90%), or the change in two consecutive steps of the data/fitting term $\mathcal{D}$ is smaller than a small number $\varepsilon_4 > 0$ (typically $\varepsilon_4 = 10^{-4}$).

8.5.3 Local Fourier analysis for the PDFP method

In this section we shall use the LFA is to analyse the smoothing properties of the PDFP iterations applied to the linearised system $\mathbf{N}_h[\overline{\boldsymbol{z}}^h]\boldsymbol{z}^h = \mathbf{G}_h[\overline{\boldsymbol{z}}^h]$ obtained by freezing coefficients in (8.37) at some outer step. Here $\boldsymbol{z}^h$ and $\overline{\boldsymbol{z}}^h$ denote the exact solution and the current approximation and $\mathbf{N}[\overline{\boldsymbol{z}}^h]$ and $\mathbf{G}[\overline{\boldsymbol{z}}^h]$ the resulting discrete operators from the linearisation at $\overline{\boldsymbol{z}}^h$.

Let $\varphi_h(\boldsymbol{\theta},\mathbf{x}) = \exp(\mathbf{i}\boldsymbol{\theta}\mathbf{x}/h)\cdot\widehat{\mathbf{I}}$ be grid functions, where $\widehat{\mathbf{I}} = (1,1,1,1,1,1)^\top$, $\boldsymbol{\theta} = (\theta_1,\theta_2)^\top \in \boldsymbol{\Theta} = (-\pi,\pi]^2$, $\mathbf{x}\in\Omega_h^\infty$, and $\mathbf{i} = \sqrt{-1}$. Similarily, our LFA is performed over the infinite grid

$$\Omega_h^\infty = \{\mathbf{x}\in\Omega | \mathbf{x} = (x_{1_i}, x_{2_j})^\top = ((2i-1)\,h/2, (2j-1)\,h/2)^\top,\ i,j\in\mathbb{Z}^2\}. \tag{8.57}$$

and applied to each grid point $\xi = (i,j)$ separately. Here we denote by

$$\bar{\mu}_{\text{loc}} = \max_{\xi\in\Omega_h} \mu_{\text{loc}}$$

the smoothing factor defined as the worst possible value of the local smoothing factor $\mu_{\text{loc}} = \mu(\xi)$ over Ω_h and $\mathbf{N}_h(\xi)\boldsymbol{z}^h(\xi) = \mathbf{G}_h(\xi)$ the local discrete system centered and defined only within a small neighborhood of ξ and $\boldsymbol{u}^h(\xi) = [u_1^h(\xi), u_2^h(\xi)]$.

Let us consider first the case of the PCGS ($\omega = 1$) approach. The splitting

$$\mathbf{N}_h(\xi) = \mathbf{N}_h^{[+]}(\xi) + \mathbf{N}_h^{[0]}(\xi) + \mathbf{N}_h^{[-]}(\xi)$$

leads the local inner iterations to

$$\mathbf{N}_h^{[+]}(\xi)\overline{\boldsymbol{z}}_{new}^h(\xi) + \mathbf{N}_h^{[0]}(\xi)\overline{\boldsymbol{z}}_{new}^h(\xi) + \mathbf{N}_h^{[-]}(\xi)\overline{\boldsymbol{z}}_{old}^h(\xi) = \mathbf{G}_h(\xi) \tag{8.58}$$

where $\overline{\boldsymbol{z}}_{old}^h(\xi)$ and $\overline{\boldsymbol{z}}_{new}^h(\xi)$ are the approximations to $\boldsymbol{z}^h(\xi)$ before and after the inner smoothing step, respectively. Here

$$\mathbf{N}_h^{[+]}(\xi) = \begin{bmatrix} -\mathcal{L}_1^{h[+]}(\xi) & 0 & 0 \\ 0 & -\mathcal{L}_2^{h[+]}(\xi) & 0 \\ 0 & 0 & -\mathcal{L}_3^{h[+]}(\xi) \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ -\alpha_1\mathcal{L}_1^{h[+]}(\xi) & 0 & 0 \\ 0 & -\alpha_1\mathcal{L}_2^{h[+]}(\xi) & 0 \\ 0 & 0 & -\alpha_2\mathcal{L}_3^{h[+]}(\xi) \end{bmatrix}, \tag{8.59}$$

$$\mathbf{N}_h^{[0]}(\xi) = \begin{bmatrix} -\mathcal{L}_1^{h[0]}(\xi) & 0 & 0 \\ 0 & -\mathcal{L}_2^{h[0]}(\xi) & 0 \\ 0 & 0 & -\mathcal{L}_3^{h[0]}(\xi) \\ \sigma_{11}(\xi) & \sigma_{12}(\xi) & \sigma_{13}(\xi) \\ \sigma_{21}(\xi) & \sigma_{22}(\xi) & \sigma_{23}(\xi) \\ \sigma_{31}(\xi) & \sigma_{32}(\xi) & \sigma_{33}(\xi) \end{bmatrix} \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \\ -\alpha_1\mathcal{L}_1^{h[0]}(\xi) & 0 & 0 \\ 0 & -\alpha_1\mathcal{L}_2^{h[0]}(\xi) & 0 \\ 0 & 0 & -\alpha_2\mathcal{L}_3^{h[0]}(\xi) \end{bmatrix}, \tag{8.60}$$

$$\mathbf{N}_h^{[-]}(\xi) = \begin{bmatrix} -\mathcal{L}_1^{h[-]}(\xi) & 0 & 0 \\ 0 & -\mathcal{L}_2^{h[-]}(\xi) & 0 \\ 0 & 0 & -\mathcal{L}_3^{h[-]}(\xi) \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ -\alpha_1\mathcal{L}_1^{h[-]}(\xi) & 0 & 0 \\ 0 & -\alpha_1\mathcal{L}_2^{h[-]}(\xi) & 0 \\ 0 & 0 & -\alpha_2\mathcal{L}_3^{h[-]}(\xi) \end{bmatrix}, \tag{8.61}$$

$$-\mathcal{L}_m^{h[+]}(\xi) = \frac{1}{h^2}\begin{bmatrix} 0 & 0 & 0 \\ -D_{m2}(\overline{z}_m(\xi)) & 0 & 0 \\ 0 & -D_{m1}(\overline{z}_m(\xi)) & 0 \end{bmatrix}, \tag{8.62}$$

$$-\mathcal{L}_m^{h[0]}(\xi) = \frac{1}{h^2}\begin{bmatrix} 0 & 0 & 0 \\ 0 & \Sigma_m(\xi) & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{8.63}$$

and

$$-\mathcal{L}_m^{h[-]}(\xi) = \frac{1}{h^2}\begin{bmatrix} 0 & -D_{m3}(\overline{z}_l(\xi)) & 0 \\ 0 & 0 & -D_{m3}(\overline{z}_l(\xi)) \\ 0 & 0 & 0 \end{bmatrix}, \tag{8.64}$$

for $m = 1, 2, 3$. Subtracting (8.58) from $\mathbf{N}_h(\xi)\boldsymbol{z}^h(\xi) = \mathbf{G}_h(\xi)$ yields the system of local error equations

$$\mathbf{N}_h^{[+]}(\xi)\overline{\boldsymbol{e}}_{new}^h(\xi) + \mathbf{N}_h^{[0]}(\xi)\overline{\boldsymbol{e}}_{new}^h(\xi) + \mathbf{N}_h^{[-]}(\xi)\overline{\boldsymbol{e}}_{old}^h(\xi) = 0$$

or

$$\overline{\boldsymbol{e}}_{new}^h(\xi) = \mathbf{S}_h(\xi)\overline{\boldsymbol{e}}_{old}^h(\xi)$$

where

$$\overline{\boldsymbol{e}}_{old}^h(\xi) = \boldsymbol{z}^h(\xi) - \overline{\boldsymbol{z}}_{old}^h(\xi) \text{ and } \overline{\boldsymbol{e}}_{new}^h(\xi) = \boldsymbol{z}^h(\xi) - \overline{\boldsymbol{z}}_{new}^h(\xi)$$

are the error functions and

$$\mathbf{S}_h(\xi) = -[\mathbf{N}_h^{[0]}(\xi) + \mathbf{N}_h^{[+]}(\xi)]^{-1}[\mathbf{N}_h^{[-]}(\xi)] \tag{8.65}$$

is the amplification factor. Hence, the PDFP local smoothing factor for this case is defined by

$$\mu_{\text{loc}} = \sup\{|\rho(\widehat{\mathbf{S}}_h(\xi, \boldsymbol{\theta}))| : \boldsymbol{\theta} \in \boldsymbol{\Theta}_{\text{high}}\} \tag{8.66}$$

where

$$\widehat{\mathbf{S}}_h(\xi, \boldsymbol{\theta}) = -[\widehat{\mathbf{N}}_h^{[0]}(\xi, \boldsymbol{\theta}) + \widehat{\mathbf{N}}_h^{[+]}(\xi, \boldsymbol{\theta})]^{-1}[\widehat{\mathbf{N}}_h^{[-]}(\xi, \boldsymbol{\theta})]$$

is the Fourier symbol of $\mathbf{S}_h(\xi)$. Recall that the Fourier symbols of $\mathcal{L}_l^{h[+]}(\xi)$ and $\mathcal{L}_l^{h[+]}(\xi)$ denoted by

$$-\mathcal{L}_m^{h[+]}(\xi, \boldsymbol{\theta}) = \frac{1}{h^2}(\Sigma_m(\xi) - D_{m1}(\xi)\exp(-\mathbf{i}\theta_1) - D_{m2}(\xi)\exp(-\mathbf{i}\theta_2)) \tag{8.67}$$

and

$$-\mathcal{L}_m^{h[-]}(\xi, \boldsymbol{\theta}) = -\frac{1}{h^2}(D_{m3}(\xi)(\exp(\mathbf{i}\theta_1) + \exp(\mathbf{i}\theta_2))). \tag{8.68}$$

are used to compute (8.66).

For the case of the $\omega-$PCGS approach, the PDFP local smoothing factor can be defined in a similar way as (8.66),

$$\mu_{\text{loc}} = \sup\{|\rho(\widehat{\mathbf{S}}_h(\xi, \boldsymbol{\theta}, \omega))| : \boldsymbol{\theta} \in \boldsymbol{\Theta}_{\text{high}}\}, \tag{8.69}$$

where the Fourier symbol of the amplification factor $\mathbf{S}_h(\xi, \omega)$ is given by

$$\widehat{\mathbf{S}}_h(\xi, \boldsymbol{\theta}, \omega) = [\widehat{\mathbf{N}}_h^{[0]}(\xi, \boldsymbol{\theta}) + \omega \widehat{\mathbf{N}}_h^{[+]}(\xi, \boldsymbol{\theta})]^{-1}[(1-\omega)\widehat{\mathbf{N}}_h^{[0]}(\xi, \boldsymbol{\theta}) - \omega \widehat{\mathbf{N}}_h^{[-]}(\xi, \boldsymbol{\theta})] \in \mathbb{C}^{6\times 6}. \tag{8.70}$$

To select the optimal value of ω, we used four registration problems shown in Figure 8.6 on a 32×32 grid. Our results indicated that $\omega = 0.7$ provides the good smoothing properties ($\overline{\mu}^*_{\text{loc}} \approx 0.60$). We also conducted several numerical tests to comfirm that (8.37) is a potential smoother for our FAS-NMG method to solve (8.24); see Table 8.1 in §8.6.2.

8.6 Numerical experiments and results

To validate and evaluate our variational registration model (8.6) and the performance of our Algorithm 8.5.2, we first perform a series of tests to verify the model effectiveness. Second, we test with respect to different resolutions. In all registration problems, the bilinear interpolation was used to compute the transformed template image $T(\boldsymbol{u})$ and we stared our MG algorithm with $\nu_1 = \nu_2 = PCGSiter = 10$, $\beta_1 = 1$, $\beta_2 = 10^{-2}$, $\boldsymbol{u}^{[0]} = 0$, and $c^{[0]} = 0$.

8.6.1 Qualify of registration

In this test, we evaluate the robustness of the proposed registration model (8.6) for the cases where the required geometric and intensity transformations are very complex.

Shown in Figure 8.6 are results from four clinical cases. In each case, the reference R and the template T are from different view points and times. Shown across each row are the reference R and template T and the registered image $T_{\boldsymbol{u}^*}$. Even in the presence of significant intensity variations, the registered images are in good agreement with the reference and show good qualitative registration results.

As shown in Figure 8.7 for results from the second problem in Figure 8.6 (d) and (e), although the registered images by two regularisation techniques for c are almost identical, the registration results in Figure 8.7 (c) and (d) confirm that $\mathcal{K}(c)$ is suitable for this hard problem where very accurate results are required for clinical image analysis; see at the white arrow locations.

Shown in Figure 8.8 are results from the third problem in Figure 8.6 (g) and (h). They indicate that our PDE-based registration model (8.24) is more robust than that of the previous work of [106] as given by (8.28). Here $\mathcal{D} = \mathcal{D}_{\text{II}}^{\text{SSD}}$, $(\mu,\ \lambda,\ \alpha_1,\ \alpha_2) = (1.00,\ 1.00,\ 0.10,\ 0.05)$ and Dirichlet boundary condition $u_l = 0$ $(l = 1, 2)$ were used through this test.

8.6.2 Multigrid performance

As is well-known, the main property of MG algorithms is that their convergence does not depend on an increasing sequence of resolutions (or a decreasing mesh parameter h). Thus, in the second test our experiments was designed to investigate this property.

To do this, we re-solve four registration problems of medical data as shown in the first and second columns in Figure 8.6 and started the registration processes with $h = 1/128,\ 1/256,\ 1/512$. Here we define a work unit used in measuring computational work as the work of performing a smoother or relaxation step on the finest grid defined as follows:

$$\begin{aligned} 1 \text{ WU} = &(\text{cost of discretisating and constructing the linearised system per grid point} \\ &+ \text{cost of PCGS updating per grid point})N \text{ (if } N \text{ is the number of grid points)} \end{aligned}$$

Thus a work unit in performing one step of our smoother can be estimated by

$$1 \text{ WU} = (177 + 284(PCGSiter))N$$

where each grid point in the linearised system (6×6) given in (8.52) is solved by the Gaussian elimination method, which have the cost of $\frac{(6)^3}{3} + \frac{(6)^2}{2} - \frac{5(6)}{6}$ additions and $\frac{(6)^3}{3} + (6)^2 - \frac{(6)}{3}$ multiplications. Therefore, the total costs of one V-cycle used L coarse grids can be estimated by

$$\text{V-cycle cost} = (\nu_1 + \nu_2)(177 + 284(PCGSiter))N\sum_{k=0}^{L}(1/4)^k \ < \ \frac{4}{3}(\nu_1 + \nu_2) \text{ WUs.}$$

Here we have ignored the cost of interpolation and restriction procedures as well as the cost of residual correction procedures because they are relative small compared with that of smoothing procedures. Recall that ν_1, ν_2, and $PCGSiter$ denote respectively the number of pre- and post-smoothing and PCGS steps.

The numerical results are reported in Table 8.1 where one can see three quantities: the numbers of MG cycles 'M'; the relative reduction of dissimilarity D = $\widetilde{\varepsilon}_3$; the work units 'WUs'.

As expected from a MG technique, Table 8.1 shows that our MG algorithm is $h-$independent. Moreover, it took only one or two MG steps to solve the registration problems and reduce the dissimilarities between the reference and registered images more than 85% for all problems.

8.7 Conclusion

This chapter introduced an improved monomodal image registration model combining a non-parametric intensity and geometric transformation, as an alternative model to using mutual information for a typical case of multimodal images where the given images have the similar features, but different intensity variations. We modelled these transformations to be non-parametric and extended the full curvature model proposed in the previous chapter to constrain them. In order to solve the resulting Euler-Lagrange system of higher order and nonlinear PDEs, we applied the idea of the PDFP method and used the LFA to analyse its smoothing

	Example 1 M/D/WUs	**Example** 2 M/D/WUs	**Example** 3 M/D/WUs	**Example** 4 M/D/WUs
	$\alpha_1 = 10^{-4}, \alpha_1 = 2\alpha_2$	$\alpha_1 = \alpha_2 = 10^{-4}$	$\alpha_1 = 10^{-4}, \alpha_1 = 2\alpha_2$	$\alpha_1 = \alpha_2 = 10^{-4}$
$h = 1/128$	1/0.0322/27	1/0.0264/27	2/0.1183/53	1/0.0914/27
$h = 1/256$	1/0.0381/27	1/0.0301/27	2/0.1242/53	1/0.0953/27
$h = 1/512$	1/0.0410/27	1/0.0357/27	2/0.1299/53	2/0.1004/53
$h = 1/128$	5/0.0321/133	5/0.0263/133	6/0.1182/160	5/0.0913/133
$h = 1/256$	5/0.0381/133	5/0.0300/133	6/0.1241/160	5/0.0952/133
$h = 1/512$	5/0.0409/133	5/0.0356/133	6/0.1298/160	6/0.1003/133

Table 8.1: Registration results of Algorithm 8.5.2 with the proposed solver in Algorithm 8.5.1 for processing four sets of clinical data shown in the first and second columns of Figure 8.6. The letters 'M', 'D', and 'WUs' mean the number of MG steps, the relative reduction of dissimilarity ($\widetilde{\varepsilon}_3$), and the work units, respectively. The last 3 rows are results for dropping the mean of relative residuals to 10^{-4}.

properties. As expected, we saw that it appears to be a potential smoother for our FAS-NMG framework and leads the associated multigrid method to be $h-$independent convergence. Numerical results showed that the proposed registration model is reliable to i) normalise image intensities between images and ii) register the given images. Moreover, they also showed that the developed multigrid method is fast and accurate in leading to visually pleasing results for practical applications.

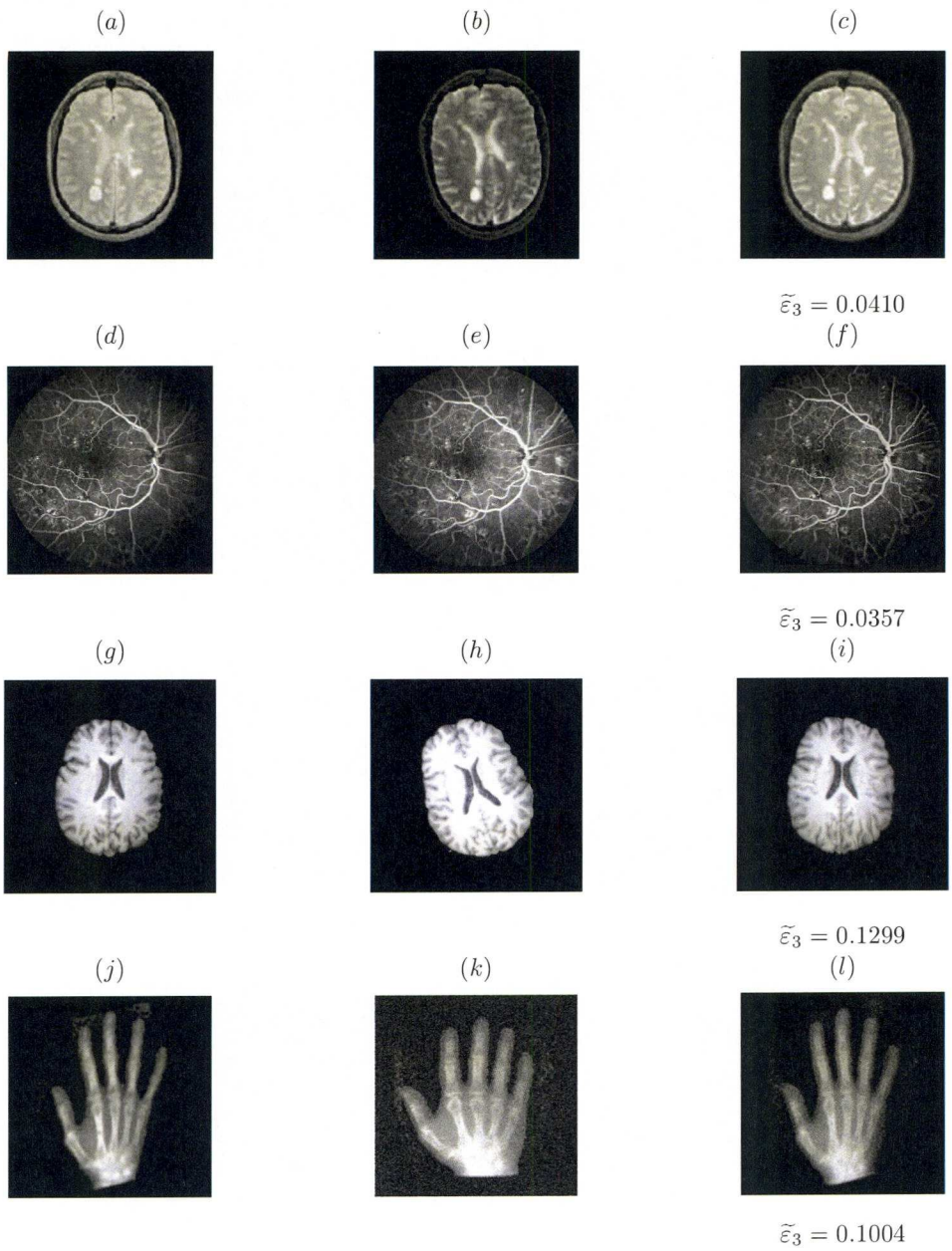


Figure 8.6: Numerical results with unknown registration. Shown in each row is the reference R, template T, and register image T_{u^*}.

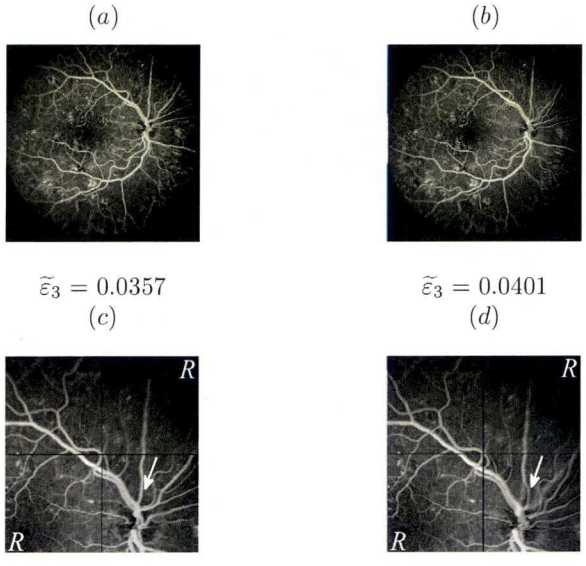


Figure 8.7: Numerical results for the second problem shown in Figure 8.6 (d) and (e) by two regularisation techniques for c. Left to right: results by $\mathcal{K}(c)$ and $\mathcal{R}^{L_2}(c)$. Top to bottom: registered images and composite views in the middle right regions between R and T_{u^*}. As shown in (c), the intensity variations of T_{u^*} in the bottom right region (at the white arrow location) by $\mathcal{K}(c)$ is well-matched, compared with those of (d) by $\mathcal{R}^{L_2}(c)$.

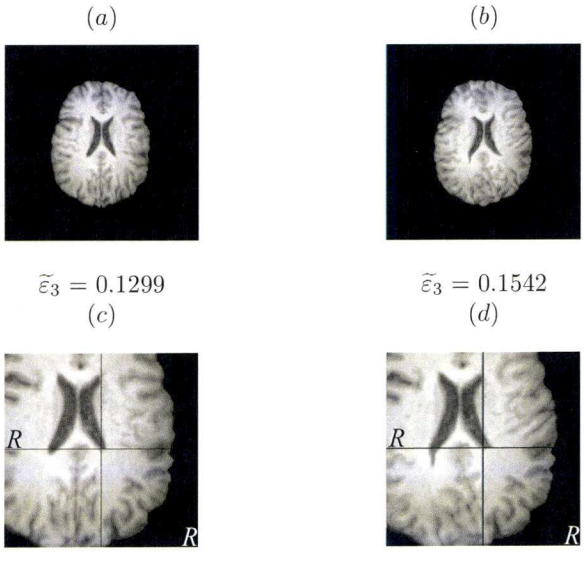


Figure 8.8: Numerical results for the third problem shown in Figure 8.6 (g) and (h) by two PDE-based registration models. Left to right: results by our model (8.24) and that of [106] (8.28). Top to bottom: registered images and composite views between R and T_{u^*}. The top right and bottom left regions of T_{u^*} in (c) by our model are well-registered with the adjacent regions of R, compared with those of (d) by [106].

Chapter 9

Summary and Future Directions

This thesis presented the author's PhD work on new contributions of four effective variational models and five efficient numerical methods for solving image registration problems.

9.1 Summary

The first contribution discussed in this thesis is a novel affine image registration model. In this work, we investigated the robustness issue of Gauss-Newton (GN) and Levenberg-Marquardt (LM) methods in solving affine image registration problems. As a result, we saw that these methods are sensitive to initial guesses and require methods of getting good initial guesses to ensure their convergence. Four existing methods were reviewed and tested. However, we found that there are always difficult cases for which these initial guesses are not sufficient. Such cases include getting pre-registration images for deformable registration problems. Then, we proposed the regularised affine registration (RAR) model that is less demanding than the standard method for initial guesses. To find the optimal regularisation parameter, we applied a coarse-to-fine approach to initialise the RAR model. Numerical results showed that the developed multilevel algorithm is generally reliable and robust in i) solving the affine image registration problems ii) providing a good initial guess for deformable models.

Second, we presented an efficient multigrid approach for variational image registration models based on the sum of squared differences (SSD) between images. A unified approach for designing fixed-point (FP) type smoothers was proposed and analysed by the local Fourier analysis (LFA) using Fischer–Modersitzki's diffusion and curvature image registration models [46, 47]. We found that the resulting linearised systems for both models are h-ellipticity and suitable with the pointwise collective Gauss-Seidel (PCGS) relaxation scheme. As expected, numerical experiments not only showed that the proposed multigrid approach is h-independent convergence, but it is also more effective than those in a large class of existing iterative methods developed by [46, 47, 48, 65, 89, 90, 131, 135, 145].

Third, we presented a novel discontinuity-preserving image registration model based on the modified total variation (TV) regularisation with the so-called potential function. As a consequence, the new model can be simply interpreted as a half way model between the diffusive

and TV regularisations for solving both smooth and non-smooth registration problems. In order to solve the resulting Euler-Lagrange system, several iterative methods including the so-called stabilising fixed-point (SFP) approach were proposed and tested using both realistic and synthetic images. As expected, we found that the SFP method can be used as a robust smoother in our nonlinear multigrid framework. Numerical experiment showed that the associated FAS-NMG method is much faster than standard unilevel methods like semi-implicit (SI) and additive operator splitting (AOS) time marching approaches in convergence and delivering the same numerical results.

Fourth, we reviewed five commonly used PDE-based models, which are effective for solving either smooth or non-smooth registration problems. Motivated by the attractive properties of the Fischer–Modersitzki's curvature registration model [47], we proposed the so-called full curvature model that appears to deliver excellent results for both registration problems. As a result, the Euler-Lagrange system of two coupled PDEs is highly nonlinear and of fourth order so standard unilevel iterative methods are not appropriate. To end this, we proposed several FP type smoothers including the SFP and primal-dual fixed-point (PDFP) methods and then used both LFA and and numerical tests to select the most effective type of smoothers which turns out to be the PDFP type smoothers. Numerical tests not only confirmed that the proposed curvature model is more robust in registration quality than previous work by [47, 48, 79, 78, 73, 75, 74], but also that the proposed multigrid method is fast and accurate in delivering visually-pleasing registration results for a wide range of applications.

Finally, we presented a new variational image registration model for monomodal images with the presence of significant intensity variations occurred in one of the given images. The new model aims to search simultaneously an optimal intensity and geometric transformation modelled to be non-parametric and constrained with the curvature model used successfully in our previous work. As a consequence, the resulting Euler-Lagrange system consisting of three nonlinear fourth order PDEs is required to solve in an efficient way. We extended the idea of the PDFP method for solving that system in a FAS-NMG framework. Numerical results confirmed the robustness of our new registration model and multigrid algorithm for solving clinical applications.

9.2 Future directions

The ideas presented in this thesis can be expended in different directions. First, the capabilities of nonlinear multigrid methods encourage us to develop fast numerical algorithms not only for multimodal image registration, but also three- and four-dimensional images. This is because most existing methods in literature are not fast enough for practical use. Second, variational models used in this thesis could be extended to include soft and/or hard constraints in order to improve registration qualities. Finally, hybrid modelling that allows simultaneously registration and segmentation precesses is also important because it can improve registration and/or segmentation results.

Bibliography

[1] E.R. Arce-Santana and A. Alba. Image registration using markov random coefficient and geometric transformation fields. *Pattern Recognition*, 42:1660–1671, 2009.

[2] G. Aubert, R. Deriche, and P. Kornprobst. Computing optical flow via variational techniques. *SIAM J. Appl. Math.*, 60(1):156–182, 1999.

[3] G. Aubert and P. Kornprobst. *Mathematical Problems in Image Processing: Partial Differential Equations and the Calculus of Variations (2nd Edition)*. Springer, 2006.

[4] G. Aubert and L. Vese. A variational method in image recovery. *SIAM J. Numer. Anal.*, 34(5):1984–1979, 1997.

[5] O. Axelsson and V.A. Barker. *Finite element solution of boundary value problems*. Academic Press, 1984.

[6] N. Badshah and K. Chen. Multigrid method for the Chan-Vese model in variational segmentation. *Communications in Computational Physics*, 4(2):294–316, 2008.

[7] N. Badshah and K. Chen. On two multigrid algorithms for modelling variational multiphase image segmentation. *IEEE Transactions on Image Processing*, 18(5):1097–1106, 2009.

[8] R. Bajcsy and S. Kovačič. Multiresolution elastic matching. *Computer vision, graphic and image processing*, 46(1):1–21, 1989.

[9] M. J. Black and A. Rangarajan. On the unification of line process, outline rejection, and robust statistics with applications in early vision. *International Journal of Computer Vision*, 19(1):57–92, 1996.

[10] M. J. Black and A. Rangarajan. The robust estimation of multiple motions: parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding*, 63(1):75–104, 1996.

[11] A. Brandt. Multi-level adaptive solutions to BVPs. *Math. Comp.*, 31:333–390, 1977.

[12] W.L. Briggs, V.E. Henson, and S.F. McCormick. *A Multigrid Tutorial (2nd Edition)*. SIAM Publications, Philadelphia, USA, 2000.

[13] C. Brito-Loeza and K. Chen. Multigrid method for a modified curvature driven diffusion model for image inpainting. *Journal of Computational Mathematics*, 26(6):856–875, 2008.

[14] C. Brito-Loeza and K. Chen. Numerical algorithms for euler's Elastica digital inpainting model. *Submitted*, 2009.

[15] C. Broit. *Optimal registration of deformed images.* PhD thesis, University of Pennsylvania, USA, 1981.

[16] L.G. Brown. A survey of image registration techniques. *ACM Comput. Sur.*, 24(4):325–376, 1992.

[17] A. Bruhn and J. Weickert. *Towards Ultimate Motion Estimation: Combining Highest Accuracy with Real-Time Performance.* in Proceeding of 10th International Conference on Computer Vision ICCV 2005, Beijing, People's Republic of China, IEEE Computer Society Press, Vol. 1, 749-755, 2005.

[18] A. Bruhn and J. Weickert. Multigrid platform for real-time motion computation with discontinuity-preserving variational methods. *International Journal of Computer Vision*, 70(3):257–277, 2006.

[19] A. Bruhn, J. Weickert, C. Feddern, T. Kohlberger, and C. Schnörr. Variational optical flow computation in real time. *IEEE Transactions on Image Processing*, 14(5):608–615, 2005.

[20] A. Bruhn, J. Weickert, T. Kohlberger, and C. Schnörr. Discontinuity-preserving computation of variational optical flow in real time. *LNCS*, 3459:279–290, 2005.

[21] G.F. Carey and R. Krishnan. On a nonlinear iterative method in applied mechanic-part 1. *Computer Methods in Applied Mechanics and Engineering*, 26:173–180, 1981.

[22] T.F. Chan and K. Chen. On a nonlinear multigrid algorithm with primal relaxation for the image total variation minimization. *Journal of Numerical Algorithms*, 41:387–411, 2006.

[23] T.F. Chan and K. Chen. An optimization-based multilevel algorithm for total variation image denoising. *Multiscale Mod. Simu.*, 5(2):615–645, 2006.

[24] T.F. Chan, K. Chen, and J.L. Carter. Iterative methods for solving the dual formulation arising from image restoration. *Electronic Transactions on Numerical Analysis*, 26:299–311, 2007.

[25] T.F. Chan, G.H. Golub, and P. Mulet. A nonlinear primal-dual method for total variationbased image restoration. *SIAM J. Sci. Comput.*, 20:1964–1997, 1999.

[26] T.F. Chan and P. Mulet. On the convergence of the lagged diffusivity fixed point method in total variation image restoration. *SIAM J. Numer. Analysis*, 36(2):354–367, 2007.

[27] T.F. Chan and J.H. Shen. *Image Processing and Analysis - Variational, PDE, Wavelet, and Stochastic Methods.* SIAM Publications, Philadelphia, USA, 2005.

[28] P. Charbonnier, L. Blanc-Féraud, G. Aubert, and M. Barlaud. Deterministic edge-preserving regularization in computed imaging. *IEEE Transactions on Image Processing*, 6(2):298–311, 1997.

[29] J. Chen, E. Haber, , and D.W. Oldenburg. Three-dimensional numerical modelling and inversion of magnetometric resistivity data. *Geophys. J. Inter.*, 149(3):679–697, 2002.

[30] K. Chen. *Matrix Preconditioning Techniques and Applications.* Cambridge University Press, UK, 2005.

[31] K. Chen and X.-C. Tai. A nonlinear multigrid method for total variation minimization from image restoration. *Journal of Scientific Computing*, 32(2):115–138, 2007.

[32] N. Chumchob and K. Chen. A robust affine image registration method. *International Journal of Numerical Analysis and Modeling*, 6(2):311–334, 2009.

[33] N. Chumchob and K. Chen. A robust multigrid approach for variational image registration models. *Submitted*, 2010.

[34] N. Chumchob and K. Chen. A variational approach for discontinuity-preserving image registration. *Submitted*, 2010.

[35] N. Chumchob, K. Chen, and C. Brito-Loeza. A fourth order variational image registration model and its fast multigrid algorithm. *Submitted*, 2010.

[36] B. Dacorogna. *Introduction to the Calculus of Varations.* Introduction to the Calculus of Varations, 2004.

[37] E. D'Afostino, F. Maes, D. Vandermeulen, and P. Suetens. A viscous fluid model for multimodal non-rigid image registration using mutual information. *Med. Imag. Ana.*, 7:565–575, 2003.

[38] E. D'Afostino, J. Modersitzki, F. Maes, D. Vandermeulen, B. Fischer, and P. Suetens. Free-form registration using mutual information and curvature regularization. *LNCS*, 2717:11–20, 2003.

[39] J.E. Dennis and R.B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations (Classics in Applied Mathematics).* Society for Industrial Mathematics, 1987.

[40] R. Deriche, P. Kornprobst, and G. Aubert. *Optical flow estimation while preserving its discontinuities: A variational approach.* in Proceeding of 2nd Asian Conference on Computer Vision, Singapore, Springer Verlag, New York, 290-295, 1997.

[41] A. Doicu, F. Schreier, and M. Hess. Iteratively regularized gauss-newton method for bound-constrained problem in atmospheric remote sensing. *Comp. Physics. Commu.*, 153:59–65, 2003.

[42] A. Doicu, F. Schreier, and M. Hess. Iteratively regularized methods for remote sensing. *J. Quan. Spec.*, 83:47–61, 2004.

[43] W. Enkelmann. Investigations of multigrid algorithms for the estimations of optical flow fields in image sequences. *Computer Vision, Graphics, and Image Processing*, 43:150–177, 1988.

[44] D.J. Eyre. *Unconditionally gradient stable time marching the Cahn-Hilliard equation.* in Computational and Mathematical models of Microstructural Evolution (1998), edited by J.W. Bullard et al. Material Research Society, Warrendale, PA, 39-46, 1998.

[45] D.J. Eyre. An unconditionally stable one-step scheme for gradient systems. *Unpublished*, 1998.

[46] B. Fischer and J. Modersitzki. Fast diffusion registration. *Contemporary Mathematics*, 313:117–129, 2002.

[47] B. Fischer and J. Modersitzki. Curvature based image registration. *Journal of Mathematical Imaging and Vision*, 18:81–85, 2003.

[48] B. Fischer and J. Modersitzki. A unified approach to fast image registration and a new curvature based registration technique. *Linear Algebra and Its Applications*, 380:107–124, 2004.

[49] B. Fischer and J. Modersitzki. Ill-posed medicine—an introduction to image registration. *Inverse Problems*, 24:(1–19), 2008.

[50] R. Fletcher. *Practical Methods of Optimization.* John Wiley and Sons, 1987.

[51] C. Frohn-Schauf, S. Henn, L. Hömke, and K. Witsch. *Total variation based image registration.* in Proceedings of the International Conference on PDE-Based Image Processing and Related Inverse Problems Series: Mathematics and Visualization, edited by X.-C. Tai, K.-A. Lie, T.F. Chan and S. Osher, Springer Verlag, 305-323, 2006.

[52] C. Frohn-Schauf, S. Henn, and K. Witsch. Nonlinear multigrid methods for total variation image denoising. *Comput. Visual. Sci.*, 7:199–206, 2004.

[53] C. Frohn-Schauf, S. Henn, and K. Witsch. Multigrid based total variation image registration. *Comput. Visual. Sci.*, 11:101–113, 2008.

[54] S. Gao, L. Zhang, H. Wang, R. de Crevoisier, D.D. Kuban, R. Mohan, and L. Dong. A deformable image registration method to handle distended rectums in prostate cancer radiotherapy. *Med. Phys.*, 33(9):3304–3312, 2006.

[55] E. Giusti. *Minimal Surfaces and Functions of Bounded Variation. Monographs in Mathematics, Vol. 80.* Birkhauser, 1984.

[56] A.A. Goshtasby. *2D and 3D Image Registration for Medical, Remote Sensing, and Industrail Applications.* John Wiley and Sons, 2005.

[57] V. Grimm, S. Henn, and K. Witsch. A higher-order pde-based image registration approach. *Numerical Linear Algebra with Applications*, 13:399–417, 2006.

[58] A. Guimond, A. Roche, N. Ayache, and J. Meunier. Three-dimensional multimodal brain wrapping using the demons algorithm and adptive intensity corrections. *IEEE Tran. Med. Imaging*, 20(1):58–69, 2001.

[59] E. Haber, S. Heldmann, and J. Modersitzki. Adaptive mesh refinement for non parametric image registration. *SIAM J. Sci. Comput.*, 30(6):3012–3027, 2008.

[60] E. Haber, S. Heldmann, and J. Modersitzki. A computational framework for image-based constrained registration. *Linear Algebra and its Application*, 431:459–470, 2009.

[61] E. Haber, R. Horesh, and J. Modersitzki. Numerical optimization for constrained image registration. *To appear in Numerical Linear Algebra with Applications*, DOI: 10.1002/nla.715, 2010.

[62] E. Haber and J. Modersitzki. Numerical solutions for volumn preserving image registration. *Inverse Problems*, 20:1621–1638, 2004.

[63] E. Haber and J. Modersitzki. *COFIR: coarse and fine image registration.* Real-Time PDE-Constrained Optimization, edited by L.T. Biegler, O. Ghattas, M. Heinkenschloss, D. Keyes and B. van Bloemen Waanders (Philadelphia, PA: SIAM), 3749, 2006.

[64] E. Haber and J. Modersitzki. Intensity gradient based registration and fusion of multimodal images. *Medical Image Computing and Computer-Assisted Intervention*, 3216:591–598, 2006.

[65] E. Haber and J. Modersitzki. A multilevel method for image registration. *SIAM J. Sci. Comput.*, 27(5):1594–1607, 2006.

[66] E. Haber and D.W. Oldenburg. A GCV based method for nonlinear ill-posed problems. *Computational Geosciences*, 4(1):41–63, 2000.

[67] W. Hackbusch. *Multi-grid Methods and Applications.* Springe-Verlag, Berlin, Heidelberg, New York, 1985.

[68] J.V. Hajnal, D.L.G. Hill, and D. Hawkes. *Medical Image Registration.* The Biomedical Engineering Series, CRC Press, 2001.

[69] S. Hamilton, M. Benzi, and E. Haber. New multigrid smoothers for the oseen problem. *To appear in Numerical Linear Algebra with Applications*, 2009.

[70] M. Hanke. A regularizing Levenberg-Marquart scheme, with applications to inverse groundwater filtration problems. *Inverse Problems*, 13:79–95, 1997.

[71] S. Heldmann, O. Mahnke, D. Potts, J. Modersitzki, and B. Fischer. *Fast computation of Mutual Information in a variational image registration approach.* Bildverarbeitung für die Medizin, edited by T. Tolxdorff, J. Braun, H. Handels, A. Horsch and H.P. Meinzer, Springer, 448-452, 2004.

[72] S. Henn. A Levenberg-Marquart scheme for nonlinear image registration. *BIT Numer. Math.*, 43:743–759, 2003.

[73] S. Henn. A multigrid method for a fourth-order diffusion equation with application to image processing. *SIAM J. Sci. Comput.*, 27(3):831–849, 2005.

[74] S. Henn. A full curature based algorithm for image registration. *J Math Imaging Vis*, 24:195–208, 2006.

[75] S. Henn. A translation and rotation invariant Gauss-Newton like scheme for image registration. *BIT Numerical Mathematics*, 46:325–344, 2006.

[76] S. Henn and K. Witsch. Iterative multigrid regularization techniques for image matching. *SIAM J. Sci. Comput.*, 23(4):1077–1093, 2001.

[77] S. Henn and K. Witsch. Multimodal image registration using a variational approach. *SIAM J. Sci. Comput.*, 25(4):1429–1447, 2003.

[78] S. Henn and K. Witsch. Image registration based on multiscale energy information. *Multiscale Modeling and Simulation*, 4(2):584–609, 2005.

[79] S. Henn and K. Witsch. A variational image registration approach based on curvature scale space. *LNCS*, 3459:143–154, 2005.

[80] G. Hermosillo, C. Chefd'Hotel, and O. Faugeras. A variational approach to multi-modal image matching. *Technical Report 4117, INRIA, France*, 2001.

[81] G. Hermosillo, C. Chefd'Hotel, and O. Faugeras. Variational methods for multimodal image matching. *International Journal of Computer Vision*, 50(3):329–343, 2002.

[82] D.L.G. Hill, P.G. Batchelor, M. Holden, and D.J. Hawkes. Medical image registration. *Phys. Med. Biol.*, 46:1–45, 2001.

[83] L. Hömke. A multigrid method for anisotropic PDEs in elastic image registration. *Numer. Linear Algebra Appl.*, 13:215–229, 2006.

[84] B. Horn and B. Schunck. Determining optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[85] M. Jenkinson, P. Bannister, M. Brady, and S. Smith. Improves optimisation for the robust and accurate linear registration and motion correction of brain images. *NeoroImage*, 17:825–841, 2002.

[86] M. Jenkinson and S. Smith. A global optimisation method for robust affine registration of brain images. *Med. Imag. Ana.*, 5:143–156, 2001.

[87] C.T. Kelly. *Iterative Methods for Optimization (Frontiers in Applied Mathematics)*. SIAM Publications, Philadelphia, USA, 1999.

[88] K. Königsberger. *Analysis 2*. Springer, 2004.

[89] H. Köstler, K. Ruhnau, and R. Wienands. Multigrid solution of the optical flow system using a combined diffusion- and curvature-based regularizer. *Numer. Linear Algebra Appl.*, 15:201–218, 2008.

[90] J. Larrey-Ruiz and J. Morales-Sánchez. Optimal parameters selection for non-parametric image registration methods. *LNCS*, 4179:564–575, 2006.

[91] J. Larrey-Ruiz, R. Verdú-Monedero, and J. Morales-Sánchez. A fourier domain framework for variational image registration. *Journal of Mathematical Imaging and Vision*, 32:57–72, 2008.

[92] P.G. Lelièvre and D.W. Oldenburg. Magnetic forrward modelling and inversion for high susceptibility. *Geophys. J. Inter.*, 166(1):76–90, 2006.

[93] T. Lu, P. Neittaanmäki, and X.-C. Tai. A parallel splitting up method and its application to Navier-Stokes equations. *Appl. Math. Lett.*, 4(2):25–29, 1991.

[94] W. Lu, M. L. Chen, G. H. Olivera, K. J. Ruchala, and T. R. Mackie. Fast free-form deformable registration via calculus of variations. *Phys. Med. Biol.*, 49:3067–3087, 2004.

[95] X. Lu, S. Zhang, H. Su, and Y. Chen. Mutual information-based multimodal image registration using a novel joint histogram estimation. *Computerized Medical Imaging and Graphics*, 32:202–209, 2008.

[96] L. Lucchese, S. Leorin, and G.M. Cortrlazzo. Estimation of two-dimensional affine transformations through polar curve matching and its application to image mosaicking and remote-sensing data registration. *IEEE Trans. Image Processing*, 15(10):3008–3018, 2006.

[97] A. Madabhushi and J.K. Udupa. New methods of MR image intensity standardization via generalized scale. *Med. Phys.*, 33(9):3426–3434, 2006.

[98] A. Madabhushi, J.K. Udupa, and G. Moonis. Comparing MR image intensity standardization against tissue characterizability of magnetization transfer ratio imaging. *Journal of Magnetic Resonance Imaging*, 24:667–675, 2006.

[99] F. Maes, D. Vandermeulen, and P. Suetens. Comparative evaluation of multiresolution optimization strategies for multimodal image registration by maximization of mutual information. *Med. Image. Anal.*, 3(4):373–386, 1999.

[100] J.B.A. Maintz and M.A. Viergever. A survey of medical image registration. *Med. Image. Anal.*, 2(1):1–36, 1998.

[101] J.V. Manjón, J.J. Lull, J. C-Caballero, and G. G-Martí. A nonparametric MRI inhomogeneity correction method. *Med. Image. Anal.*, 11(4):336–345, 2007.

[102] J.L. Marrroquin, B.C. Vemuri, S. Botello, F. Calderon, and A. F.-Bouzas. An accurate and efficient baysian model for automatic segmetation of brain MRI. *IEEE Trans. Med. Imaging*, 21(8):934–945, 2002.

[103] A.R. Mitchell and D.F. Griffiths. *The Finite Difference Method in Partial Differential Equations.* John Wiley and Sons, Chichester, 1980.

[104] J. Modersitzki. *Numerical Methods for Image Registration.* Oxford, 2004.

[105] J. Modersitzki. *FAIR: Flexible Algorithms for Image Registration.* SIAM Publications, Philadelphia, USA, 2009.

[106] J. Modersitzki and S. Wirtz. Combining homogenization and registration. *LNCS*, 4057:257–263, 2006.

[107] H. Nam, R.A. Renaut, K. Chen, H. Guo, and G.E. Farin. Improved inter-modality image registration using normalized mutual information with coarse-binned histograms. *Communications in Numerical Methods in Engineering*, 25:583–595, 2009.

[108] G. A Newman and G M. Hoversten. Solution strategies for two- and three-dimensional electromagnetic inverse problems. *Inverse Problems*, 16:1357–1375, 2000.

[109] J. Nocedal and S.J. Wright. *Numerical Optimization.* Springer-Verlag, NewYork, 1999.

[110] L.G. Nyúl and J.K. Udupa. On standardizing the MR image intensity scale. *Magnetic Resonance in Medicine*, 42:1072–1081, 1999.

[111] W. Ou and C. Chefd'Hotel. *Polynomial Intensity Correction for Multimodal Image Registration.* Proceedings of the Sixth IEEE international conference on Symposium on Biomedical Imaging: From Nano to Macro, Boston, Massachusetts, USA, 939-942, 2009.

[112] R.L. Parker. *Geophysical Inverse Theory.* Princeton University Press, 1994.

[113] S. Periaswamy and H. Farid. Elastic registration in the presence of intensity variations. *IEEE Trans. Med. Imaging.*, 22(7):865–874, 2003.

[114] J.P.W. Pluim, J.B.A. Maintz, and M.A. Viergever. Image registration by maximization of combined mutual information and gradient information. *IEEE Trans. Med. Imaging.*, 19(8):809–814, 2000.

[115] T. Pock, M. Urschler, C. Zach, R. Beichel, and H. Bischof. A duality based algorithm for *TV*-L^1-optical-flow image registration. *LNCS*, 4792:511–518, 2007.

[116] J.F. Ralph. *Automatic Image Matching (AIM) GUI Toolset.* Department of Electrical Engineering and Electronics, The University of Liverpool, 2006.

[117] W.C. Rheinboldt. *Methods for Solving Systems of Nonlinear Equations (2nd Edition).* SIAM Publications, Philadelphia, USA, 1998.

[118] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Phys.D*, 60:259–268, 1992.

[119] D. Rueckert, M.J. Bowden, D.J.G. Hill, and D.J. Hawkes. Image template matching using mutual information and NP-windows. *Medical Imaging*, 2:438–447, 2000.

[120] D. Rueckert, L.I. Sonoda, C. Hayes, D.L.G. Hill, M.O. Leach, and D.J. Hawkes. Nonrigid registration using free-form deformations: Application to breast MR images. *IEEE Trans. Medical Imaging*, 18(8):712–721, 1999.

[121] Y. Saad. *Iterative Methods for Sparse Linear Systems (2nd Edition).* SIAM Publications, Philadelphia, USA, 2003.

[122] J. Savage and K. Chen. An improved and accelerated nonlinear multigrid method for total-variation denoising. *International Journal of Computer Mathematics*, 82(8):1001–1015, 2005.

[123] J. Savage and K. Chen. *On multigrids for solving a class of improved total variation based staircasing reduction models.* in Proceedings of the International Conference on PDE-Based Image Processing and Related Inverse Problems Series: Mathematics and Visualization, edited by X.-C. Tai, K.-A. Lie, T.F. Chan and S. Osher, Springer Verlag, 69-94, 2006.

[124] O. Scherzer. *Mathematical Models for Registration and Applications to Medical Imaging.* Springer, New York, 2006.

[125] O. Schmitt, J. Modersitzki, S. Heldmann, and S. Wirtz. Image registration of sectioned brain. *Intl. J. Comp. Vision.*, 73(1):5–39, 2007.

[126] B. Seynaeve, E. Rosseel, B. Nicolaï, and S. Vandewalle. Fourier mode analysis of multigrid methods for partial differential equations with random coefficients. *Journal of Computational Physics*, 224:132–149, 2007.

[127] B.W. Silverman. *Density Estimation for Statistics and Data Analysis.* Chapman and Hall, London, New York, 1986.

[128] G.D. Smith. *Numerical Solution of Partial Differential Equations: Finite Difference Methods.* Clarendon Press, Oxford, 1985.

[129] E. Spedicato and M.T. Vespucci. Numerical experiments with variations of the Gauss-Newton algorithm for nonlinear least squares. *J. Opt. Theory and Appl.*, 57(2):323–339, 1988.

[130] C. Studholme, D.L.G. Hill, and D.J. Hawkes. Automated $3-\mathrm{D}$ registration of MR and CT images of the head. *Med. Image. Anal.*, 1(2):163–175, 1996.

[131] M. Stürmer, H. Köstler, and U. Rüde. A fast full multigrid solver for applications in image processing. *Numer. Linear Algebra with Appl.*, 15:187–200, 2008.

[132] P. Thévenaz, U. E. Ruttimann, and M. Unser. A pyramid approach to subpixel registration based on intensity. *IEEE Trans. Image Processing*, 7(1):27–41, 1998.

[133] J.P. Thirion. Image mathching as diffusion process: an analogy with maxwell's demons. *Med. Image Anal.*, 2(3):243–260, 1998.

[134] U. Trottenberg, C. Oosterlee, and A. Schuller. *Multigrid.* Academic Press, 2001.

[135] R. Verdú-Monedero and J. Larrey-Ruiz. Frequency implementation of the euler–lagrange equations for variational image registration. *IEEE Signal Processing Letters*, 15:321–324, 2008.

[136] C. R. Vogel and M. E. Oman. Iterative methods for total variation denoising. *SIAM Journal on Scientific Computing*, 17(1):227–238, 1996.

[137] C.R. Vogel. *Computational Methods for Inverse Problems.* SIAM Publications, Philadelphia, USA, 2002.

[138] J. Weickert, B.M. ter Haar Romeny, and M. Viergever. Effcient and reliable schemes for nonlinear diffusion filtering. *IEEE Trans. Image Processing*, 7:398–410, 1998.

[139] P. Wesseling. *Multigrid Methods.* Edwards: Philadelphis, PA, U.S.A., 2004.

[140] R. Wienands and W. Joppich. *Practical Fourier Analysis for Multigrid Method.* Chapman & Hall/CRC, U.S.A., 2005.

[141] M. Xia and B. Liu. Image registration by "super-curves". *IEEE Trans. Image Processing*, 13(5):720–732, 2004.

[142] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime $TV\text{-}L^1$ optical flow. *J. of Magnetic Resonance Imaging*, 4713:214–223, 2007.

[143] P. Zhilkin and M.E. Alexander. Affine registration: a comparison of several programs. *J. of Magnetic Resonance Imaging*, 22:55–66, 2004.

[144] Y. Zhuge, J.K. Udupa, J. Liu, and P.K. Saha. Image background inhomogeneity correction in MRI via intensity standardization. *Computerized Medical Imaging and Graphics*, 33(1):7–16, 2009.

[145] D. Zikic, W. Wein, and A. Khamene. Fast deformable registration of 3D−ultrasound data using a variational approach. *LNCS*, 4190:915–923, 2006.