Computational Methods in Applied Mathematics | Volume 20: Issue 1

An Effective Diffeomorphic Model and Its Fast Multigrid Algorithm for Registration of Lung CT Images

Tony Thompson and Ke Chen

DOI: https://doi.org/10.1515/cmam-2018-0126 | Published online: 01 Feb 2019

https://doi.org/10.1515/cmam-2018-0126

Volume 20, Issue 1, Pages 141–168, 2019

Abstract

Image registration is the process of aligning sets of similar, but different, intensity image functions to track changes between the images. In medical image problems involving lung images, variational registration models are a very powerful tool which can aid in effective treatment of various lung conditions and diseases. However a common drawback of many variational models, such as the diffusion model [19] and even optic flow models [8, 22], is the lack of control of folding in the deformations leading to physically inaccurate transformations. For this reason, such models are generally not suitable for real life lung imaging problems where folding cannot occur.

There are two approaches offering reliable solutions (though not necessarily accurate). The first approach is a parametric model such as the affine registration model, still widely used in many applications, but it cannot track local changes or yield accurate results. The second approach is to impose an extra constraint on the transformation of registration as in the work by [11, 36, 48], at the cost of increased nonlinearity. An alternative to the second approach, achieving diffeomorphic transforms without adding any constraints, is an inverse consistent model such as by Christensen-Johnson [15] from computing explicitly both the forward and inverse transforms. However one must deal with the strong non-linearity in the formulation.

In this paper we first propose a simplified inverse consistent model to avoid the inclusion of strong non-linearities and then a fast non-linear multigrid (NMG) technique to overcome the extra computational work required by the inverse consistent model. Experiments, performed on real medical CT images, show that our proposed inverse consistent model is robust to both parameter choice and non-folding in the transformations when compared with diffusion type models.

Keywords. System of nonlinear PDEs, Existence, Image registration, Diffeomorphic map, Fast multigrid solver.

1 Introduction

A challenge which frequently arises in a lot of real world applications, and especially in medical imaging, is image registration. An image registration technique works by fixing one image in a pair or set of similar images to be the 'reference' image and then applying geometric transformations to the remaining image/s, called the 'template' image/s, with the goal of aligning the template image/s with the reference image. The important role that registration plays in many aspects of medical imaging problems can be seen in recent works of [1, 16, 25, 27, 33]. Especially in diagnostics of lung problems [12, 17, 26, 28, 40, 44], registration tasks such as motion correction and feature tracking are routinely carried out and any increase in accuracy is highly desirable in improving patient care. Since the transformations within lung images are in general highly non-uniform, non-parametric models such as [6, 7, 9–11] are typically favoured over parametric models such as [3, 18, 34, 37]. Our main concern is this former type.

Denoting by $R, T \in \Omega \subset \mathbb{R}^d$ respectively a reference function and template image function, we are looking to determine the transformation $\boldsymbol{\varphi}(\boldsymbol{x}, \boldsymbol{u})$ such that

$$T\left(\boldsymbol{\varphi}(\boldsymbol{x}, \boldsymbol{u})\right) \equiv T\left(\boldsymbol{x} + \boldsymbol{u}\right) \equiv T_{\boldsymbol{u}} \approx R \equiv R(\boldsymbol{x}) \text{ for } \boldsymbol{x} = (x_1, \ldots, x_d)^T \in \Omega \subset \mathbb{R}^d \qquad (1.1)$$

*Centre for Mathematical Imaging Techniques and Department of Mathematical Sciences, University of Liverpool, United Kingdom. Emails: [anthony.thompson, k.chen]@liv.ac.uk

†Corresponding author. Web: http://www.liv.ac.uk/∼cmchenke

where $\boldsymbol{\varphi}(\boldsymbol{x},\boldsymbol{u}) = \boldsymbol{x}+\boldsymbol{u}(\boldsymbol{x})$ and $\boldsymbol{u} \equiv \boldsymbol{u}(\boldsymbol{x}) = (u_1(\boldsymbol{x}),\dots,u_d(\boldsymbol{x}))^T$ denotes the displacement field. Throughout the remainder of this paper we will only consider the two-dimensional case $d = 2$, however the ideas presented are extendible to the three-dimensional case $d = 3$. In addition, we will also assume that the image domain Ω is given by the unit square $\Omega = [0,1]^2$.

We can formulate the variational image registration problem mathematically in the following way. The task of finding the transformation $\boldsymbol{\varphi}$ is equivalent to that of determining the displacement field $\boldsymbol{u}$, which is achieved by solving a minimisation problem of the following form

$$\min_{\boldsymbol{u}} E(\boldsymbol{u}) = \mathscr{D}\left(R,T,\boldsymbol{u}\right) + \alpha\mathscr{R}\left(\boldsymbol{u}\right) \tag{1.2}$$

where $E(\boldsymbol{u})$ denotes some general energy functional, $\mathscr{D}$ is some dissimilarity measure of T,R, $\mathscr{R}$ is a regularisation term required to constrain $\boldsymbol{u}$ and overcome the ill-posedness of the problem and $\alpha \in \mathbb{R}^+$ is some weighting parameter. For the purposes of this paper, we will assume that R,T are mono-modal images, and as a result the common choice of dissimilarity measure is the sum of squared distances (SSD), although this is not the only possible choice [39]. The SSD term is given by the following

$$\mathscr{D}\left(R,T,\boldsymbol{u}\right) = \frac{1}{2}\int_{\Omega} \left|T_{\boldsymbol{u}} - R\right|^2 \, d\Omega \tag{1.3}$$

where $|\cdot|$ denotes the Euclidean norm and $T_{\boldsymbol{u}} \equiv T(\boldsymbol{x}+\boldsymbol{u})$. Moreover, there is a large choice of regularisation term [2,5,20,23,38]. Here we shall mainly consider one of these, $\mathscr{R}(\boldsymbol{u}) = \|\nabla\boldsymbol{u}\|^2 = \|u_1\|^2 + \|u_2\|^2$, in order to focus on the idea of diffeomorphism of $\boldsymbol{\varphi}$. Unfortunately energy functionals of the form shown in (1.2), in general, do not avoid the potential problem of mesh folding in the transformation $\boldsymbol{\varphi}$. Since we are considering real life medical imaging problems, a transformation with folding would suggest that the transformation is physically inaccurate and therefore incorrect. One mathematical solution to overcome this problem is to impose the nonlinear constraint $Q_{min} = \min\det(\nabla\boldsymbol{\varphi}) > 0$ as done in recent works of [11,36,48] and in particular the term $\min(\det(\nabla\boldsymbol{\varphi}) - 1)^4/(\det(\nabla\boldsymbol{\varphi}))^2$ is added in [11].

However, we consider here another solution to this folding problem by extending the model (1.2) to include an additional term, explicitly linking the forward transform $\boldsymbol{\varphi}$ and the inverse transform $\boldsymbol{\psi}$ between T,R, which enforces the transformation $\boldsymbol{\varphi}$ to be inverse consistent and therefore non-folding. A simple way to ensure diffeomorphism is for the transformation $\boldsymbol{\varphi}$ and its inverse $\boldsymbol{\psi}$ to satisfy the relation $\boldsymbol{\varphi} = \boldsymbol{\psi}^{-1}$ since $\boldsymbol{\varphi}\circ\boldsymbol{\varphi}^{-1} = \boldsymbol{\psi}\circ\boldsymbol{\psi}^{-1} = \boldsymbol{I}\boldsymbol{x} = \boldsymbol{x}$ where $\boldsymbol{I}$ denotes the identity mapping. The first variant including an inverse consistency constraint (and $\boldsymbol{\varphi}$ only) leads to a minimisation problem of the form

$$\min_{\boldsymbol{u}} E^{(I)}(\boldsymbol{u}) = \mathscr{D}\left(R,T,\boldsymbol{u}\right) + \alpha\mathscr{R}\left(\boldsymbol{u}\right) + \beta\mathscr{I}\left(\boldsymbol{\varphi}\left(\boldsymbol{x},\boldsymbol{u}\right), \boldsymbol{\varphi}^{-1}\left(\boldsymbol{x},\tilde{\boldsymbol{u}}\right)\right) \tag{1.4}$$

where $\mathscr{I}$ denotes the inverse consistency constraint, $\boldsymbol{\varphi}^{-1}$, $\tilde{\boldsymbol{u}}$ denote the inverses of $\boldsymbol{\varphi}$, $\boldsymbol{u}$ respectively and $0 \le \beta \in \mathbb{R}$ is a second weighting parameter. There are different choices for the inverse consistency constraint [14,15,17,34]. In this paper however we consider the second variant of an inverse consistent model, using both $\boldsymbol{\varphi}$ and $\boldsymbol{\psi}$, with the following form

$$\begin{aligned}\min_{\boldsymbol{u},\boldsymbol{v}} E^{(II)}\left(\boldsymbol{u},\boldsymbol{v}\right) = \frac{1}{2}\int_{\Omega} & \mathscr{D}\left(R,T,\boldsymbol{u}\right) + \mathscr{D}\left(T,R,\boldsymbol{v}\right) + \alpha\big(\mathscr{R}\left(\boldsymbol{u}\right) + \mathscr{R}\left(\boldsymbol{v}\right)\big) \\ & + \beta\Big(\mathscr{I}\big(\boldsymbol{\varphi}\left(\boldsymbol{x},\boldsymbol{u}\right), \boldsymbol{\psi}^{-1}\left(\boldsymbol{x},\tilde{\boldsymbol{v}}\right)\big) + \mathscr{I}\big(\boldsymbol{\psi}\left(\boldsymbol{x},\boldsymbol{v}\right), \boldsymbol{\varphi}^{-1}\left(\boldsymbol{x},\tilde{\boldsymbol{u}}\right)\big)\Big)\, d\Omega.\end{aligned} \tag{1.5}$$

where $\mathscr{D}\left(T,R,\boldsymbol{v}\right)$, $\mathscr{R}\left(\boldsymbol{v}\right)$ and $\mathscr{I}\big(\boldsymbol{\psi}\left(\boldsymbol{x},\boldsymbol{v}\right), \boldsymbol{\varphi}^{-1}\left(\boldsymbol{x},\tilde{\boldsymbol{u}}\right)\big)$ denote the similarity measure, regularisation term and inverse consistency constraint respectively for the backward problem $R \to T$, also where $\boldsymbol{v}$, $\boldsymbol{\psi}$ denote the backward displacement and transformation respectively with $\tilde{\boldsymbol{v}}$, $\boldsymbol{\psi}^{-1}$ denoting their inverses. We aim to simplify this second variant and propose an efficient multigrid numerical scheme.

The remainder of this paper will be set out as follows. In §2 we will introduce the Christensen-Johnson model based on (1.5), as well as our proposed simplification to avoid additional non-linearities when compared with general diffusion type models, in addition to our proposed numerical approach. Next in §3 we will introduce our fast NMG scheme to overcome the increased computational cost resulting from the additional work required by the model, before showing some experimental results on real medical CT images in §4. Finally in §5 we will present our conclusions.

2 A simplified inverse consistent model and its algorithm

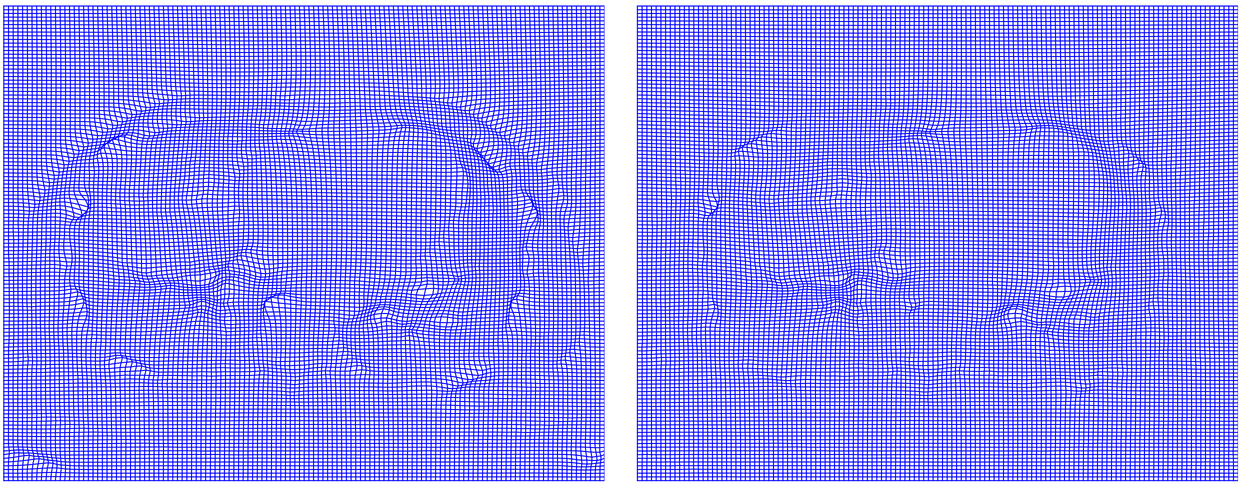
Several authors have discussed similar registration models for two images to symmetrically deform toward one another in multiple passes [14, 29, 42, 47]. The realization of a diffeomorphic transform is achieved by working with 4 deformation fields instead of 1. Here we follow the work by Christensen-Johnson [15] who proposed a model to overcome the problem of non-inverse consistent transformations by using only 2 deformation fields. The model satisfies our requirement of having a more physically accurate transformation robust to folding. They achieved this through a combination of two things: (i) A term was added into the standard form of the energy functional shown in (1.2) to impose inverse consistency and take on the form show in (1.5); (ii) The forward $(T \to R)$ and backward $(R \to T)$ registration problems were computed simultaneously. These things, combined with a SSD dissimilarity term (1.3) and diffusion regularisation term, led to the formation of their inverse consistent model which is given by the following

$$\begin{aligned}\min_{\boldsymbol{u},\boldsymbol{v}} E^{IC}(\boldsymbol{u},\boldsymbol{v}) = \frac{1}{2}\int_\Omega |T_{\boldsymbol{u}} - R|^2 + |R_{\boldsymbol{v}} - T|^2 + \alpha\left(|\nabla \boldsymbol{u}|^2 + |\nabla \boldsymbol{v}|^2\right) \\ + \beta\left(\left|\boldsymbol{\varphi}(\boldsymbol{x},\boldsymbol{u}) - \boldsymbol{\psi}^{-1}(\boldsymbol{x},\tilde{\boldsymbol{v}})\right|^2 + \left|\boldsymbol{\psi}(\boldsymbol{x},\boldsymbol{v}) - \boldsymbol{\varphi}^{-1}(\boldsymbol{x},\tilde{\boldsymbol{u}})\right|^2\right) d\Omega \qquad (2.1)\end{aligned}$$

where $|\cdot|$ denotes the F-norm for matrices (reduced to modulus for scalar quantities), $\boldsymbol{\varphi}, \boldsymbol{\psi}$ denote the forward and backward transformations, $\boldsymbol{\varphi}^{-1}, \boldsymbol{\psi}^{-1}$ denote the inverse transformations, $\boldsymbol{u}, \boldsymbol{v}$ denote the forward and backward displacements and $\tilde{\boldsymbol{u}}, \tilde{\boldsymbol{v}}$ denote the inverse displacements respectively. The full minimisation problem was then split into two sub-problems corresponding to the forward and backward registration problems respectively. This resulted in (2.1) being written in the following way

$$\begin{cases} \min\limits_{\boldsymbol{u}} E_1^{IC}(\boldsymbol{u},\boldsymbol{v}) = \dfrac{1}{2}\displaystyle\int_\Omega |T_{\boldsymbol{u}} - R|^2 + \alpha|\nabla \boldsymbol{u}|^2 + \beta|\boldsymbol{u} - \tilde{\boldsymbol{v}}|^2 \, d\Omega, \quad \tilde{\boldsymbol{v}}(\boldsymbol{x}) = \boldsymbol{\psi}^{-1}(\boldsymbol{x}) - \boldsymbol{x} \\ \min\limits_{\boldsymbol{v}} E_2^{IC}(\boldsymbol{u},\boldsymbol{v}) = \dfrac{1}{2}\displaystyle\int_\Omega |R_{\boldsymbol{v}} - T|^2 + \alpha|\nabla \boldsymbol{v}|^2 + \beta|\boldsymbol{v} - \tilde{\boldsymbol{u}}|^2 \, d\Omega, \quad \tilde{\boldsymbol{u}}(\boldsymbol{x}) = \boldsymbol{\varphi}^{-1}(\boldsymbol{x}) - \boldsymbol{x}. \end{cases} \qquad (2.2)$$

Noting that the constraints in (2.2) are respectively $\boldsymbol{\psi}(\tilde{\boldsymbol{v}}(\boldsymbol{x})) = \boldsymbol{x} - \boldsymbol{\psi}$ and $\boldsymbol{\varphi}(\tilde{\boldsymbol{u}}(\boldsymbol{x})) = \boldsymbol{x} - \boldsymbol{\varphi}$ i.e. $\boldsymbol{\psi}(\tilde{\boldsymbol{v}}(\boldsymbol{x})) + \boldsymbol{v} = \mathbf{0}$, $\boldsymbol{\varphi}(\tilde{\boldsymbol{u}}(\boldsymbol{x})) + \boldsymbol{u} = \mathbf{0}$, the explicit computation of them is a difficult and computationally expensive task owing to their non-linear nature. However, this kind of model is effective at preventing mesh folding as is illustrated in Figure 1 where the mesh problem on the left is fixed by the model on the right plot.



(a) Bad mesh of the transformation $\boldsymbol{\varphi}$ obtained from the standard diffusion model: $Q_{min} = -0.245$

(b) Good mesh obtained from the new inverse consistent model: $Q_{min} = 0.114$

Figure 1: Comparison of two registration meshes for Example 2 as shown in Figure 2 for the same parameters $\alpha = \frac{1}{25}$ and $\beta = 10^4$ (See §4).

We are motivated to overcome the difficulty of computing the inverse displacements $\tilde{\boldsymbol{u}}$ and $\tilde{\boldsymbol{v}}$ directly. We propose to replace these terms with linear approximations. This simplification allows us to remove the additional non-linearities from the inverse consistent terms, leaving only the non-linearities seen in diffusion type models, while still retaining the advantages of the inverse consistent model. We know that the transformations $\boldsymbol{\varphi}, \boldsymbol{\psi}$, and their inverses $\boldsymbol{\varphi}^{-1}, \boldsymbol{\psi}^{-1}$, should satisfy the following relations $\boldsymbol{\varphi}^{-1}(\boldsymbol{\varphi}(\boldsymbol{x}, \boldsymbol{u})) = \boldsymbol{x}$, $\boldsymbol{\psi}^{-1}(\boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{v})) = \boldsymbol{x}$. Expanding out leads to the following equalities

$$\begin{cases} \boldsymbol{\varphi}^{-1}(\boldsymbol{\varphi}(\boldsymbol{x}, \boldsymbol{u})) = \boldsymbol{\varphi}(\boldsymbol{x}, \boldsymbol{u}) + \tilde{\boldsymbol{u}}(\boldsymbol{\varphi}(\boldsymbol{x}, \boldsymbol{u})) = \boldsymbol{x} + \boldsymbol{u}(\boldsymbol{x}) + \tilde{\boldsymbol{u}}(\boldsymbol{x} + \boldsymbol{u}(\boldsymbol{x})) = \boldsymbol{x} \\ \boldsymbol{\psi}^{-1}(\boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{v})) = \boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{v}) + \tilde{\boldsymbol{v}}(\boldsymbol{\psi}(\boldsymbol{x}, \boldsymbol{v})) = \boldsymbol{x} + \boldsymbol{v}(\boldsymbol{x}) + \tilde{\boldsymbol{v}}(\boldsymbol{x} + \boldsymbol{v}(\boldsymbol{x})) = \boldsymbol{x} \end{cases} \tag{2.3}$$

which can be reduced to

$$\boldsymbol{u}(\boldsymbol{x}) + \tilde{\boldsymbol{u}}(\boldsymbol{x} + \boldsymbol{u}(\boldsymbol{x})) = 0, \qquad \boldsymbol{v}(\boldsymbol{x}) + \tilde{\boldsymbol{v}}(\boldsymbol{x} + \boldsymbol{v}(\boldsymbol{x})) = 0 \tag{2.4}$$

by using a Taylor expansion on the arguments of $\tilde{\boldsymbol{u}}, \tilde{\boldsymbol{v}}$ in (2.4), we can obtain the approximations

$$\tilde{\boldsymbol{u}}(\boldsymbol{x} + \boldsymbol{u}(\boldsymbol{x})) \approx \tilde{\boldsymbol{u}}(\boldsymbol{x}), \qquad \tilde{\boldsymbol{v}}(\boldsymbol{x} + \boldsymbol{v}(\boldsymbol{x})) \approx \tilde{\boldsymbol{v}}(\boldsymbol{x}). \tag{2.5}$$

From substituting (2.5) into (2.4), we get

$$\boldsymbol{u}(\boldsymbol{x}) \approx -\tilde{\boldsymbol{u}}(\boldsymbol{x}), \qquad \boldsymbol{v}(\boldsymbol{x}) \approx -\tilde{\boldsymbol{v}}(\boldsymbol{x}) \tag{2.6}$$

and using (2.6) in (2.1), we have

$$\begin{aligned} \min_{\boldsymbol{u}, \boldsymbol{v}} E^{IC}(\boldsymbol{u}, \boldsymbol{v}) = \frac{1}{2} \int_\Omega & |T_{\boldsymbol{u}} - R|^2 + |R_{\boldsymbol{v}} - T|^2 + \alpha \left(|\nabla \boldsymbol{u}|^2 + |\nabla \boldsymbol{v}|^2 \right) \\ & + \beta \left(|\boldsymbol{u} + \boldsymbol{v}|^2 + |\boldsymbol{v} + \boldsymbol{u}|^2 \right) d\Omega \\ \equiv g^{IC} & (\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}, \nabla \boldsymbol{u}, \nabla \boldsymbol{v}) \end{aligned} \tag{2.7}$$

which results in the following split formulation by alternating minimization

$$\begin{cases} \min_{\boldsymbol{u}} E_1^{IC}(\boldsymbol{u}, \boldsymbol{v}) = \dfrac{1}{2} \displaystyle\int_\Omega |T_{\boldsymbol{u}} - R|^2 + \alpha |\nabla \boldsymbol{u}|^2 + \beta |\boldsymbol{u} + \boldsymbol{v}|^2 \, d\Omega, \\ \\ \min_{\boldsymbol{v}} E_2^{IC}(\boldsymbol{u}, \boldsymbol{v}) = \dfrac{1}{2} \displaystyle\int_\Omega |R_{\boldsymbol{v}} - T|^2 + \alpha |\nabla \boldsymbol{v}|^2 + \beta |\boldsymbol{v} + \boldsymbol{u}|^2 \, d\Omega. \end{cases} \tag{2.8}$$

Comparing this model with (2.1), we see that we now no longer need to compute the inverse displacements $\tilde{\boldsymbol{u}}$ and $\tilde{\boldsymbol{v}}$ directly, instead we need only use the displacements $\boldsymbol{u}$ and $\boldsymbol{v}$.

To solve the minimisation problem (2.8), a discretise-optimise approach (for details see [38, 39]) was used originally, however we instead propose to use an optimise-discretise approach in addition to a fast NMG framework. This approach involves solving the Euler-Lagrange (EL) equations corresponding to (2.8), and can be shown to be given by

$$-\alpha \Delta u_m + F_m(\boldsymbol{u}, \boldsymbol{v}) = 0, \qquad -\alpha \Delta v_m + G_m(\boldsymbol{u}, \boldsymbol{v}) = 0 \tag{2.9}$$

with respective Neumann boundary conditions $\nabla u_m \cdot \boldsymbol{n} = 0$, $\nabla v_m \cdot \boldsymbol{n} = 0$, where

$$\begin{aligned} F_m(\boldsymbol{u}, \boldsymbol{v}) &= \beta (u_m + v_m) + \partial_{u_m} T_{\boldsymbol{u}} (T_{\boldsymbol{u}} - R), \\ G_m(\boldsymbol{u}, \boldsymbol{v}) &= \beta (v_m + u_m) + \partial_{v_m} R_{\boldsymbol{v}} (R_{\boldsymbol{v}} - T) \end{aligned} \tag{2.10}$$

denote respectively the force terms for component $m = 1, 2$.

We remark that the models by [14, 29, 42, 47], though involving more unknown fields to compute, can also be advantageous when the underlying deformation between T and R is large (and by design the 4 fields can be small or could be said to be half sized); in this case, it will be of interest to develop fast multigrid methods for them.

2.1 Existence of a solution for model (2.7)

Now we will prove the existence of solutions for the model (2.7) following the idea of [11] for a similar proof in a related but different model. Given the energy functional $E^{IC}(\boldsymbol{u},\boldsymbol{v})$ defined in (2.7), we wish to show that the solutions $\boldsymbol{u}^*, \boldsymbol{v}^*$ exist such that $E^{IC}(\boldsymbol{u}^*,\boldsymbol{v}^*)$ becomes minimal. We use the so called direct method [21] as in [11], consisting of the following steps:

(i) Take the minimising sequences $\{\boldsymbol{u}_n,\boldsymbol{v}_n\}$ for E^{IC}.

(ii) Show that the sequences $\{\boldsymbol{u}_n,\boldsymbol{v}_n\}$ admit subsequences $\{\boldsymbol{u}_{n_k},\boldsymbol{v}_{n_k}\}$ that converge to a solution $(\boldsymbol{u}^*,\boldsymbol{v}^*)\in\chi$ in the weak topology, where χ denotes some function space.

(iii) Show that the energy functional E^{IC} is lower semi-continuous.

Before outlining the proof, we first review some necessary theory which will be used shortly. First we introduce three assumptions which will be used for the remainder of this proof:

- **A1**: Assume that $\alpha=\beta=2$ for simplicity.
- **A2**: Assume that the image domain Ω has a C^1 boundary that is denoted by $\partial\Omega$.
- **A3**: Assume that $R,T\in C^2$.

Second, define the function space χ by the following

$$\chi := W^{1,2}\left(\Omega,\mathbb{R}^2\right)\times W^{1,2}\left(\Omega,\mathbb{R}^2\right) \tag{2.11}$$

equipped with the norm $\|(\boldsymbol{u},\boldsymbol{v})\|_\chi = \|\boldsymbol{u}\|_{W^{1,2}(\Omega,\mathbb{R}^2)}\times\|\boldsymbol{v}\|_{W^{1,2}(\Omega,\mathbb{R}^2)}$.

Remark 2.1. *Here we remark that the function space χ is reflexive, this means that there exist subsequences which converge in the weak topology. Or, in other words, given the bounded sequences $(x_n,y_n)\in\chi$ then there exist subsequences x_{n_k},y_{n_k} such that $\Phi(x_{n_k},y_{n_k})\to\Phi(x_n,y_n)$ $\forall\Phi\in\chi$.*

Third, define the following admissible sets

$$\begin{aligned}\mathcal{A} &= \left\{\boldsymbol{u}\in\mathcal{A}_0\colon \left|\int_\Omega \boldsymbol{u}(\boldsymbol{x})\,d\Omega\right|\le vol(\Omega)\,(M+diam(\Omega))\right\}\\ \mathcal{B} &= \left\{\boldsymbol{v}\in\mathcal{B}_0\colon \left|\int_\Omega \boldsymbol{v}(\boldsymbol{x})\,d\Omega\right|\le vol(\Omega)\,(N+diam(\Omega))\right\}\end{aligned} \tag{2.12}$$

where $\mathcal{A}_0=\left\{\boldsymbol{u}\in W^{1,2}\left(\Omega,\mathbb{R}^2\right)\right\}$, $\mathcal{B}_0=\left\{\boldsymbol{v}\in W^{1,2}\left(\Omega,\mathbb{R}^2\right)\right\}$ and $M,N\in\mathbb{R}$ are some constants.

Definition 2.1 (Generalised Poincaré Inequality)**.** Let $p\in[1,\infty]$ and Ω be a bounded connected open subset of $\mathbb{R}^N$ with a Lipschitz boundary, then there exists some constant $C\in\mathbb{R}$ which depends only on p and Ω so that for every function $\boldsymbol{u}\in W^{1,2}(\Omega)$

$$\|\nabla\boldsymbol{u}\|_{L^p(\Omega)}\ge C\,\|\boldsymbol{u}-\boldsymbol{u}_\Omega\|_{L^p(\Omega)} \tag{2.13}$$

where $\boldsymbol{u}_\Omega=\frac{1}{|\Omega|}\int_\Omega\boldsymbol{u}\,d\Omega$.

Lemma 2.1 (General Lower Semi-Continuity)**.** *In the image domain $\Omega\in\mathbb{R}^2$, suppose that $f\colon\Omega\to\mathbb{R}^2\times\mathbb{R}^N\to[0,\infty)$ is a continuously differentiable function and $f(\cdot,\boldsymbol{y},\boldsymbol{\xi})$ is measurable for every $(\boldsymbol{y},\boldsymbol{\xi})\in\mathbb{R}^2\times\mathbb{R}^N$. Also assume that $f(\boldsymbol{x},\boldsymbol{y},\cdot)$ is convex and that*

$$\boldsymbol{y}_n\to\boldsymbol{y}\ \textit{in}\ L^p\left(\Omega,\mathbb{R}^2\right)\ \textit{for}\ p\ge1;\qquad \boldsymbol{\xi}_n\to\boldsymbol{\xi}\ \textit{in}\ L^p\left(\Omega,\mathbb{R}^N\right)\ \textit{for}\ p\ge1. \tag{2.14}$$

Then the following result holds

$$\liminf_{n\to\infty}\int_\Omega f\left(\boldsymbol{x},\boldsymbol{y}_n(\boldsymbol{x}),\boldsymbol{\xi}_n(\boldsymbol{x})\right)\,d\Omega\ge\int_\Omega f\left(\boldsymbol{x},\boldsymbol{y}(\boldsymbol{x}),\boldsymbol{\xi}(\boldsymbol{x})\right)\,d\Omega \tag{2.15}$$

Lemma 2.2 (Coercity Condition)**.** *Let the assumptions A1 and A3 from earlier hold, then the inverse consistent model (2.7) satisfies the coercity condition. That is, there exist constants $0<C,\ K\in\mathbb{R}$ such that $\forall\boldsymbol{u}\in\mathcal{A}$, $\boldsymbol{v}\in\mathcal{B}$ the following inequality holds*

$$E^{IC}(\boldsymbol{u},\boldsymbol{v})\ge K+C\left(\|\boldsymbol{u}\|^2_{W^{1,2}(\Omega,\mathbb{R}^2)}+\|\boldsymbol{v}\|^2_{W^{1,2}(\Omega,\mathbb{R}^2)}\right) \tag{2.16}$$

where $\mathcal{A}$, $\mathcal{B}$ are the admissible sets defined in (2.12).

Proof. Suppose that we have some arbitrary transformations $\boldsymbol{u} \in \mathcal{A}$, $\boldsymbol{v} \in \mathcal{B}$, then we have

$$\begin{aligned} E^{IC}(\boldsymbol{u},\boldsymbol{v}) &= \int_\Omega \frac{1}{2}\left(|T_{\boldsymbol{u}}-R|^2+|R_{\boldsymbol{v}}-T|^2\right)+\|\nabla\boldsymbol{u}\|^2+\|\nabla\boldsymbol{v}\|^2+|\boldsymbol{u}+\boldsymbol{v}|^2+|\boldsymbol{v}+\boldsymbol{u}|^2\;d\Omega \\ &\geq \int_\Omega \|\nabla\boldsymbol{u}\|^2+\|\nabla\boldsymbol{v}\|^2\;d\Omega \end{aligned} \tag{2.17}$$

since $\frac{1}{2}|T_{\boldsymbol{u}}-R|^2 \geq 0$, $\frac{1}{2}|R_{\boldsymbol{v}}-T|^2 \geq 0$, $|\boldsymbol{u}+\boldsymbol{v}|^2 \geq 0$, $|\boldsymbol{v}+\boldsymbol{u}|^2 \geq 0$. Then, as a result of assumption $A2$, we can use the generalised Poincaré inequality (Definition 2.1) to get

$$\|\nabla\boldsymbol{u}\|^2_{L^2} \geq C_1\|\boldsymbol{u}\|^2_{L^2} - C_1|\Omega|\left(\frac{1}{|\Omega|}\left|\int_\Omega \boldsymbol{u}\,d\Omega\right|\right)^2 \tag{2.18}$$

where $C_1 \in \mathbb{R}$ is some constant. Since we know that $\boldsymbol{u} \in \mathcal{A}$ and $\left|\int_\Omega \boldsymbol{u}\,d\Omega\right| \leq vol(\Omega)\,(M+diam(\Omega))$, then we also know that there exists some $K_1 \in \mathbb{R}$ such that

$$\|\nabla\boldsymbol{u}\|^2_{L^2} \geq K_1 + C_1\|\boldsymbol{u}\|^2_{L^2} \tag{2.19}$$

using an analogous argument, and the fact that $\boldsymbol{v} \in \mathcal{B}$ and $\left|\int_\Omega \boldsymbol{v}\,d\Omega\right| \leq vol(\Omega)\,(N+diam(\Omega))$, we can show that there exist $C_2, K_2 \in \mathbb{R}$ such that the following inequality holds

$$\|\nabla\boldsymbol{v}\|^2_{L^2} \geq K_2 + C_2\|\boldsymbol{v}\|^2_{L^2}\,. \tag{2.20}$$

Then introducing the new constants C, $K \in \mathbb{R}$, and combining (2.17)-(2.20), we get

$$E^{IC}(\boldsymbol{u},\boldsymbol{v}) \geq K + C\left(\|\boldsymbol{u}\|^2_{W^{1,2}(\Omega,\mathbb{R}^2)}+\|\boldsymbol{v}\|^2_{W^{1,2}(\Omega,\mathbb{R}^2)}\right) \tag{2.21}$$

and so the coercity condition holds. □

Finally, in order for a solution to the inverse consistent model (2.7) to exist, the following existence theorem must hold

Theorem 2.3. *Given that the assumptions* $A1$*-*$A3$ *hold, then the model* (2.7) *with energy functional* $E^{IC}(\boldsymbol{u},\boldsymbol{v})$ *possesses at least one minimiser* $(\boldsymbol{u}^*,\boldsymbol{v}^*)$, $\boldsymbol{u}^* \in \mathcal{A}, \boldsymbol{v}^* \in \mathcal{B}$.

Proof. We begin by constructing the minimising sequences $\{\boldsymbol{u}_n,\boldsymbol{v}_n\}$ such that $\lim\limits_{n\to\infty} E^{IC}(\boldsymbol{u}_n,\boldsymbol{v}_n) = \inf\limits_{\boldsymbol{u}\in\mathcal{A},\boldsymbol{v}\in\mathcal{B}} E^{IC}(\boldsymbol{u},\boldsymbol{v})$ given that the energy functional E^{IC} is positive and has a lower bound 0. Moreover, the energy functional $E^{IC}(\boldsymbol{x},\boldsymbol{x})$ is finite. Then, using Lemma 2.2, for each n we have

$$M \geq E^{IC}(\boldsymbol{u}_n,\boldsymbol{v}_n) \geq K + C\left(\|\boldsymbol{u}\|^2_{W^{1,2}(\Omega,\mathbb{R}^2)}+\|\boldsymbol{v}\|^2_{W^{1,2}(\Omega,\mathbb{R}^2)}\right) \tag{2.22}$$

and so the sequences $\{\boldsymbol{u}_n,\boldsymbol{v}_n\}$ are bounded in the function space χ. Since we know that the function space χ is reflexive (Remark 2.1), then this implies that there exist some subsequences $\{\boldsymbol{u}_{n_k},\boldsymbol{v}_{n_k}\}$ which converge to $(\boldsymbol{u}^*,\boldsymbol{v}^*)$ in the weak topology. Now we see that $(\boldsymbol{u}_{n_k},\boldsymbol{v}_{n_k}) \to (\boldsymbol{u}^*,\boldsymbol{v}^*)$ in the space $W^{1,2}$ implies that $(\boldsymbol{u}_{n_k},\boldsymbol{v}_{n_k}) \to (\boldsymbol{u}^*,\boldsymbol{v}^*)$ in the L^2 space owing to the fact that $W^{1,2}$ is compactly embedded in the L^2 space i.e. $W^{1,2} \subset\subset L^2$. From assumption $\mathcal{A}2$ we know that the function g^{IC}, defined in (2.7), is convex for fixed $\boldsymbol{x},\boldsymbol{u},\boldsymbol{v},$ continuously differentiable and measurable in $\boldsymbol{x}$ for fixed $(\boldsymbol{u},\boldsymbol{v},\nabla\boldsymbol{u},\nabla\boldsymbol{v}) \in \mathbb{R}^2\times\mathbb{R}^2\times\mathbb{R}^{2\times2}\times\mathbb{R}^{2\times2}$. Therefore, using Lemma 2.1, we can say that the functional E^{IC} is weakly lower semi-continuous. That is

$$\lim_{n\to\infty}\inf\int_\Omega g^{IC}(\boldsymbol{x},\boldsymbol{u}_{n_k},\boldsymbol{v}_{n_k},\nabla\boldsymbol{u}_{n_k},\nabla\boldsymbol{v}_{n_k})\;d\Omega \geq \int_\Omega g^{IC}(\boldsymbol{x},\boldsymbol{u},\boldsymbol{v},\nabla\boldsymbol{u},\nabla\boldsymbol{v})\;d\Omega \tag{2.23}$$

thus we have

$$\inf_{\boldsymbol{u}\in\mathcal{A},\boldsymbol{v}\in\mathcal{B}} E^{IC}(\boldsymbol{u},\boldsymbol{v}) = \lim_{n\to\infty} E^{IC}(\boldsymbol{u}_{n_k},\boldsymbol{v}_{n_k}) \geq E^{IC}(\boldsymbol{u}^*,\boldsymbol{v}^*) \geq \inf_{\boldsymbol{u}\in\mathcal{A},\boldsymbol{v}\in\mathcal{B}} E^{IC}(\boldsymbol{u},\boldsymbol{v}). \tag{2.24}$$

Therefore, the solution $(\boldsymbol{u}^*,\boldsymbol{v}^*)$ is a minimiser of the energy functional E^{IC}. □

Remark 2.2. *Here we note that this proof can also be used to show the existence of solutions for the original Christensen-Johnson model* (2.1) *using a slight modification in* (2.17).

2.2 Discretisation of the inverse consistent model (2.9)

To solve the system of EL equations (2.9), we look to obtain a numerical approximation. We do this by discretising the image domain Ω^h into a uniform $n \times n$ mesh, with interval width $h = \frac{1}{n-1}$, and then using a finite difference (FD) method.

Remark 2.3. *In general we need not discretise Ω^h using a square mesh, and can instead be discretised using a $n \times m$ mesh where $n \neq m$. However it is common for lung CT slices to be square, and for this reason we work with a square mesh (by taking $m = n$).*

Doing this, as well as using a lexicographic ordering of the discrete grid points (i,j), we obtain the following discrete versions of (2.9)

$$-\alpha\big(\Delta^h u_m^h\big)_k + \big(F_m(\boldsymbol{u}^h, \boldsymbol{v}^h)\big)_k = 0, \qquad -\alpha\big(\Delta^h v_m^h\big)_k + \big(G_m(\boldsymbol{u}^h, \boldsymbol{v}^h)\big)_k = 0 \tag{2.25}$$

where

$$\big(\Delta^h u_m^h\big)_k \approx \frac{1}{h^2}\Big(\big(u_m^h\big)_{k-n} + \big(u_m^h\big)_{k-1} + \big(u_m^h\big)_{k+1} + \big(u_m^h\big)_{k+n} \Big) \tag{2.26}$$

and similar for $\big(\Delta^h v_m^h\big)_k$, also with the following discrete force terms

$$\begin{aligned}
\big(F_m(\boldsymbol{u}^h, \boldsymbol{v}^h)\big)_k &= \beta\Big(\big(u_m^h\big)_k + \big(v_m^h\big)_k\Big) + \big(\partial_{u_m}^h T_{\boldsymbol{u}}^h\big)_k\Big(\big(T_{\boldsymbol{u}}^h\big)_k - \big(R^h\big)_k\Big), \\
\big(G_m(\boldsymbol{u}^h, \boldsymbol{v}^h)\big)_k &= \beta\Big(\big(v_m^h\big)_k + \big(u_m^h\big)_k\Big) + \big(\partial_{v_m}^h R_{\boldsymbol{v}}^h\big)_k\Big(\big(R_{\boldsymbol{v}}^h\big)_k - \big(T^h\big)_k\Big)
\end{aligned} \tag{2.27}$$

where

$$\begin{aligned}
&\big(\partial_{u_1}^h T_{\boldsymbol{u}}^h\big)_k \approx \frac{1}{2h}\Big(\big(T_{\boldsymbol{u}}^h\big)_{k+1} - \big(T_{\boldsymbol{u}}^h\big)_{k-1} \Big), \big(\partial_{u_2}^h T_{\boldsymbol{u}}^h\big)_k \approx \frac{1}{2h}\Big(\big(T_{\boldsymbol{u}}^h\big)_{k+n} - \big(T_{\boldsymbol{u}}^h\big)_{k-n} \Big), \\
&\big(\partial_{v_1}^h R_{\boldsymbol{v}}^h\big)_k \approx \frac{1}{2h}\Big(\big(R_{\boldsymbol{v}}^h\big)_{k+1} - \big(R_{\boldsymbol{v}}^h\big)_{k-1} \Big), \big(\partial_{v_2}^h R_{\boldsymbol{v}}^h\big)_k \approx \frac{1}{2h}\Big(\big(R_{\boldsymbol{v}}^h\big)_{k+n} - \big(R_{\boldsymbol{v}}^h\big)_{k-n} \Big)
\end{aligned} \tag{2.28}$$

for $m = 1, 2$, $k = (j-2)(n-1) + (i-1)$ and $i, j = 2, \dots, n-1$.

There are a lot of choices of methods to solve the discrete system of equations (2.25). Some examples include the Newton method, the time-marching method and the additive operator splitting (AOS) method. However for highly non-linear equations, like the ones in (2.25), it can be difficult to ensure these methods converge to a solution. Moreover, for large images, using such methods to solve (2.25) on a single level is extremely expensive computationally. Also owing to the inverse consistent model requiring the simultaneous computation of the forward and backward problems, this expense is doubled. This problem is very common in variational models, and as such there has been a lot of research into the development of NMG methods with the purpose of greatly reducing CPU cost in solving such problems [19, 24, 30–32, 43]. In particular we note the work done by Chumchob-Chen in [19] where they developed a robust NMG framework for diffusion type models (though their model cannot avoid mesh folding).

Now we propose to use a similar NMG framework applied to our inverse consistent model. In addition we will also perform a more accurate analysis of the NMG scheme compared to that presented in [19], in order to obtain a better measure of what is required to achieve optimal convergence for the NMG scheme.

2.3 A non-linear multigrid framework

Here we will present our NMG framework based upon [19]. Multigrid methods stem from two key observations

O1: Iterative solvers, such as the Gauss-Seidel method, are effective at removing (smoothing) high frequency error components within a small number of iterations. Low frequency error components dominate convergence rates.

O2: Smooth errors (low frequency) are well approximated on coarser grids. Coarser grids have less unknowns making it feasible to do a larger number of iterations without increasing the overall cost.

By using these observations, we can restrict our problem on a fine grid to that of a much coarser grid, by alternating both smoothing and coarsening steps. On this very coarse grid, we are able to obtain a much more accurate approximation in significantly less time. From this accurate approximation, we can interpolate back up to our original fine grid to obtain an approximation to the original problem. Now we briefly outline our proposed 'full approximation scheme' NMG (FAS-NMG) algorithm (See [4] for details) within the two-grid setting. We begin by denoting the original fine grid by Ω^h and the coarse grid by Ω^H with intervals $h = \frac{1}{n-1}$ and $H = 2h$ respectively. Next we write the PDEs from (2.25) using the following operator notation

$$\boldsymbol{\mathcal{N}}_1^h[\boldsymbol{u}^h, \boldsymbol{v}^h] = \boldsymbol{\mathcal{G}}_1^h, \qquad \boldsymbol{\mathcal{N}}_2^h[\boldsymbol{u}^h, \boldsymbol{v}^h] = \boldsymbol{\mathcal{G}}_2^h \tag{2.29}$$

where $\boldsymbol{\mathcal{N}}_m^h$ and $\boldsymbol{\mathcal{G}}_m^h$ $(m = 1, 2)$ are sized 2 vectors consisting of the non-linear LHS and initial zero RHS of the discrete EL equations (2.25) for $\boldsymbol{u}^h, \boldsymbol{v}^h$ respectively. Then the FAS-NMG framework, in the two-grid setting, is as followed

---

Algorithm 1 $\left[\boldsymbol{u}_h^{(k+1)}, \boldsymbol{v}_h^{(k+1)}\right] \leftarrow \text{FAS-NMG}(R^h, T^h, n, h, \boldsymbol{u}_h^{(k)}, \boldsymbol{v}_h^{(k)}, \boldsymbol{\mathcal{G}}_1^h, \boldsymbol{\mathcal{G}}_2^h, \alpha, \nu_1, \nu_2)$

1: Pre-smoothing step by performing ν_1 steps to update $\boldsymbol{u}_h$ $\quad \bar{\boldsymbol{u}}_h^{(k)} \leftarrow \text{SMOOTH}(R^h, T^h, \boldsymbol{u}_h^{(k)}, \boldsymbol{\mathcal{G}}_1^h, \alpha, \nu_1)$

2: and then $\boldsymbol{v}_h$ $\quad \bar{\boldsymbol{v}}_h^{(k)} \leftarrow \text{SMOOTH}(R^h, T^h, \boldsymbol{v}_h^{(k)}, \boldsymbol{\mathcal{G}}_2^h, \alpha, \nu_1)$

3: Coarse-grid correction

  Compute the residuals $\boldsymbol{r}_{1\,h}^{(k)} = \boldsymbol{\mathcal{G}}_1^h - \boldsymbol{\mathcal{N}}_1^h[\boldsymbol{u}_h^{(k)}, \bar{\boldsymbol{v}}_h^{(k)}]$, $\boldsymbol{r}_{2\,h}^{(k)} = \boldsymbol{\mathcal{G}}_2^h - \boldsymbol{\mathcal{N}}_2^h[\boldsymbol{v}_h^{(k)}, \bar{\boldsymbol{u}}_h^{(k)}]$

  Restrict residuals and smooth approximations $\boldsymbol{r}_{m\,H}^{(k)} = \mathcal{R}_h^H \boldsymbol{r}_{m\,h}^{(k)}$, $\bar{\boldsymbol{u}}_H^{(k)} = \mathcal{R}_h^H \bar{\boldsymbol{u}}_h^{(k)}$, $\bar{\boldsymbol{v}}_H^{(k)} = \mathcal{R}_h^H \bar{\boldsymbol{v}}_h^{(k)}$

  Set $H = 2h$

  Form RHS of coarse grid PDEs $\boldsymbol{\mathcal{G}}_1^H = \boldsymbol{r}_1^H + \boldsymbol{\mathcal{N}}_1^H[\bar{\boldsymbol{u}}_H^{(k)}, \bar{\boldsymbol{v}}_H^{(k)}]$, $\boldsymbol{\mathcal{G}}_2^H = \boldsymbol{r}_2^H + \boldsymbol{\mathcal{N}}_2^H[\bar{\boldsymbol{u}}_H^{(k)}, \bar{\boldsymbol{v}}_H^{(k)}]$

4: Solve to obtain solutions $\boldsymbol{u}_H^{(k)}$, $\boldsymbol{v}_H^{(k)}$ to high accuracy using a coarsest grid solver.

  Compute the corrections $\boldsymbol{e}_{1\,H}^{(k)} = \boldsymbol{u}_H^{(k)} - \bar{\boldsymbol{u}}_H^{(k)}$, $\boldsymbol{e}_{2\,H}^{(k)} = \boldsymbol{v}_H^{(k)} - \bar{\boldsymbol{v}}_H^{(k)}$

  Interpolate the corrections to original fine grid level $\boldsymbol{e}_{1\,h}^{(k)} = \mathcal{I}_H^h \boldsymbol{e}_{1\,H}^{(k)}$, $\boldsymbol{e}_{2\,h}^{(k)} = \mathcal{I}_H^h \boldsymbol{e}_{2\,H}^{(k)}$

  Update current grid level approximations using correction $\hat{\boldsymbol{u}}_h^{(k)} = \bar{\boldsymbol{u}}_h^{(k)} + \boldsymbol{e}_{1\,h}^{(k)}$, $\hat{\boldsymbol{v}}_h^{(k)} = \bar{\boldsymbol{v}}_h^{(k)} + \boldsymbol{e}_{2\,h}^{(k)}$

5: Post-smoothing step by performing ν_2 steps (relaxation sweeps) $\boldsymbol{u}_h^{(k+1)} \leftarrow \text{SMOOTH}(R^h, T^h, \hat{\boldsymbol{u}}_h^{(k)}, \boldsymbol{\mathcal{G}}_1^h, \alpha, \nu_1)$

6: $\quad \boldsymbol{v}_h^{(k+1)} \leftarrow \text{SMOOTH}(R^h, T^h, \hat{\boldsymbol{v}}_h^{(k)}, \boldsymbol{\mathcal{G}}_2^h, \alpha, \nu_1)$

This Algorithm 1 can be refined on its coarse grid to recursively interact with increasingly coarser grids until a desired level is reached (e.g. 8×8), thus leading to the full v-cycle scheme. Out of the three main steps in the NMG framework (smoothing, coarse grid solver, correction), the smoothing step is the most crucial to the convergence of the scheme. As was highlighted by **O2**, only 'smooth' errors can be approximated on a coarser grid, thus any remaining high frequency error components can no longer be removed once the problem has been restricted to a coarser grid (where high frequency error components form the fine grid are not present or visible) which in turn means the NMG will take longer to converge as well as being less accurate.

2.4 Three collective pointwise smoothers for (2.25)

Here we will present three different smoother schemes to be used in our NMG scheme.

First Pointwise Smoother (S1): For our first smoother we consider the simplest type of smoother scheme to solve the system (2.25), namely we use each equation to update each displacement independently. We do this by using the following fixed point iteration scheme

$$-\alpha\big(\Delta^h u_m^h\big)_k^{(l+1)} + \big(F_m(\boldsymbol{u}^h, \boldsymbol{v}^h)\big)_k^{(l+1)} = 0, \qquad -\alpha\big(\Delta^h v_m^h\big)_k^{(l+1)} + \big(G_m(\boldsymbol{u}^h, \boldsymbol{v}^h)\big)_k^{(l+1)} = 0 \tag{2.30}$$

where

$$\begin{aligned}
\big(F_1(\boldsymbol{u}^h, \boldsymbol{v}^h)\big)_k^{(l+1)} &= \beta\Big(\big(u_1^h\big)_k^{(l+1)} + \big(v_1^h\big)_k^{(l)}\Big) \\
&\quad - \big(\partial_{u_1}^h T^h(x_1 + u_1^{(l)}, x_2 + u_2^{(l)})\big)_k \Big(\big(T^h(x_1 + u_1^{(l+1)}, x_2 + u_2^{(l)})\big)_k - \big(R^h(x_1, x_2)\big)_k\Big), \\
\big(F_2(\boldsymbol{u}^h, \boldsymbol{v}^h)\big)_k^{(l+1)} &= \beta\Big(\big(u_2^h\big)_k^{(l+1)} + \big(v_2^h\big)_k^{(l)}\Big) \\
&\quad - \big(\partial_{u_2}^h T^h(x_1 + u_1^{(l)}, x_2 + u_2^{(l)})\big)_k \Big(\big(T^h(x_1 + u_1^{(l)}, x_2 + u_2^{(l+1)})\big)_k - \big(R^h(x_1, x_2)\big)_k\Big),
\end{aligned}$$

$$
\begin{aligned}
\big(G_1(\boldsymbol{u}^h,\boldsymbol{v}^h)\big)_k^{(l+1)} &= \beta\Big(\big(v_1^h\big)_k^{(l+1)} + \big(u_1^h\big)_k^{(l)}\Big) \\
&\quad - \big(\partial^h_{v_1} R^h(x_1+v_1^{(l)}, x_2+v_2^{(l)})\big)_k \Big(\big(R^h(x_1+v_1^{(l+1)}, x_2+v_2^{(l)})\big)_k - \big(T^h(x_1,x_2)\big)_k\Big), \\
\big(G_2(\boldsymbol{u}^h,\boldsymbol{v}^h)\big)_k^{(l+1)} &= \beta\Big(\big(v_2^h\big)_k^{(l+1)} + \big(u_2^h\big)_k^{(l)}\Big) \\
&\quad - \big(\partial^h_{v_2} R^h(x_1+v_1^{(l)}, x_2+v_2^{(l)})\big)_k \Big(\big(R^h(x_1+v_1^{(l)}, x_2+v_2^{(l+1)})\big)_k - \big(T^h(x_1,x_2)\big)_k\Big).
\end{aligned} \tag{2.31}
$$

Now in order to deal with the non-linearities in the force terms of (2.30), we use the same treatment as that used in [19], namely we linearise the force terms using first order Taylor expansions. Replacing the non-linear force terms in (2.30), with their first order approximations, leads to the following smoother scheme at step (l) to update the $(l+1)$ terms

$$
\begin{cases}
\quad -\alpha\big(\Delta^h u_m^h\big)_k^{(l+1)} + \beta\Big(\big(u_m^h\big)_k^{(l+1)} + \big(v_m^h\big)_k^{(l)}\Big) \\
\qquad + \big(\partial^h_{u_m} T^h_{\boldsymbol{u}}\big)_k^{(l)} \Big[\big(T^h_{\boldsymbol{u}}\big)_k^{(l)} + \Big(\big(u_m^h\big)_k^{(l+1)} - \big(u_m^h\big)_k^{(l)}\Big)\big(\partial^h_{u_m} T^h_{\boldsymbol{u}}\big)_k^{(l)} - \big(R^h\big)_k\Big] = 0, \\
\quad -\alpha\big(\Delta^h v_m^h\big)_k^{(l+1)} \beta\Big(\big(v_m^h\big)_k^{(l+1)} + \big(u_m^h\big)_k^{(l)}\Big) \\
\qquad + \big(\partial^h_{v_m} R^h_{\boldsymbol{v}}\big)_k^{(l)} \Big[\big(R^h_{\boldsymbol{v}}\big)_k^{(l)} + \Big(\big(v_m^h\big)_k^{(l+1)} - \big(v_m^h\big)_k^{(l)}\Big)\big(\partial^h_{v_m} R^h_{\boldsymbol{v}}\big)_k^{(l)} - \big(T^h\big)_k\Big] = 0
\end{cases} \tag{2.32}
$$

for $m = 1, 2$. In order to compute the $(l+1)$ terms in (2.32), we use a lexicographic Gauss-Seidel (GSLEX) based method.

Second Pointwise Smoother (S2): Following the smoother proposed by Chumchob-Chen [19], for our second proposed smoother, we will fully couple all 4 PDEs together by using a similar scheme to (2.30) and new fixed point lineralizations as follows

$$
\begin{aligned}
\big(F_1(\boldsymbol{u}^h,\boldsymbol{v}^h)\big)_k^{(l+1)} &= \beta\Big(\big(u_1^h\big)_k^{(l+1)} + \big(v_1^h\big)_k^{(l+1)}\Big) \\
&\quad - \big(\partial^h_{u_1} T^h(x_1+u_1^{(l)}, x_2+u_2^{(l)})\big)_k \Big(\big(T^h(x_1+u_1^{(l+1)}, x_2+u_2^{(l+1)})\big)_k - \big(R^h(x_1,x_2)\big)_k\Big), \\
\big(F_2(\boldsymbol{u}^h,\boldsymbol{v}^h)\big)_k^{(l+1)} &= \beta\Big(\big(u_2^h\big)_k^{(l+1)} + \big(v_2^h\big)_k^{(l+1)}\Big) \\
&\quad - \big(\partial^h_{u_2} T^h(x_1+u_1^{(l)}, x_2+u_2^{(l)})\big)_k \Big(\big(T^h(x_1+u_1^{(l+1)}, x_2+u_2^{(l+1)})\big)_k - \big(R^h(x_1,x_2)\big)_k\Big), \\
\big(G_1(\boldsymbol{u}^h,\boldsymbol{v}^h)\big)_k^{(l+1)} &= \beta\Big(\big(v_1^h\big)_k^{(l+1)} + \big(u_1^h\big)_k^{(l+1)}\Big) \\
&\quad - \big(\partial^h_{v_1} R^h(x_1+v_1^{(l)}, x_2+v_2^{(l)})\big)_k \Big(\big(R^h(x_1+v_1^{(l+1)}, x_2+v_2^{(l+1)})\big)_k - \big(T^h(x_1,x_2)\big)_k\Big), \\
\big(G_2(\boldsymbol{u}^h,\boldsymbol{v}^h)\big)_k^{(l+1)} &= \beta\Big(\big(v_2^h\big)_k^{(l+1)} + \big(u_2^h\big)_k^{(l+1)}\Big) \\
&\quad - \big(\partial^h_{v_2} R^h(x_1+v_1^{(l)}, x_2+v_2^{(l)})\big)_k \Big(\big(R^h(x_1+v_1^{(l+1)}, x_2+v_2^{(l+1)})\big)_k - \big(T^h(x_1,x_2)\big)_k\Big).
\end{aligned} \tag{2.33}
$$

Next we linearise the force terms (2.33) by applying Taylor approximations to the discrete force terms (2.33), we obtain the following smoother scheme to update the $(l+1)$ terms at step (l)

$$
\begin{cases}
-\alpha\big(\Delta^h u_s^h\big)_k^{(l+1)} + \beta\left(\big(u_s^h\big)_k^{(l+1)} + \big(v_s^h\big)_k^{(l+1)}\right) \\
+\big(\partial^h_{u_s} T^h_{\boldsymbol{u}}\big)_k^{(l)} \Big[\big(T^h_{\boldsymbol{u}}\big)_k^{(l)} + \left(\big(u_s^h\big)_k^{(l+1)} - \big(u_s^h\big)_k^{(l)}\right)\big(\partial^h_{u_s} T^h_{\boldsymbol{u}}\big)_k^{(l)} + \left(\big(u_t^h\big)_k^{(l+1)} - \big(u_t^h\big)_k^{(l)}\right)\big(\partial^h_{u_t} T^h_{\boldsymbol{u}}\big)_k^{(l)}\Big] = 0, \\
-\alpha\big(\Delta^h v_s^h\big)_k^{(l+1)} + \beta\left(\big(v_s^h\big)_k^{(l+1)} + \big(u_s^h\big)_k^{(l+1)}\right) \\
+\big(\partial^h_{v_s} R^h_{\boldsymbol{v}}\big)_k^{(l)} \Big[\big(R^h_{\boldsymbol{v}}\big)_k^{(l)} + \left(\big(v_s^h\big)_k^{(l+1)} - \big(v_s^h\big)_k^{(l)}\right)\big(\partial^h_{v_s} R^h_{\boldsymbol{v}}\big)_k^{(l)} + \left(\big(v_t^h\big)_k^{(l+1)} - \big(v_t^h\big)_k^{(l)}\right)\big(\partial^h_{v_t} R^h_{\boldsymbol{v}}\big)_k^{(l)}\Big] = 0
\end{cases} \tag{2.34}
$$

for $s, t = 1, 2$ and $s \neq t$. Similar to **S1**, we use a GSLEX based method on (2.34) to update the $(l+1)$ terms.

Third Pointwise Smoother (S3): The above 4×4 system which must be solved at every discrete interior point in (2.34) is computationally expensive. For this reason we propose an alternate, simplified version of **S2** while still maintaining some coupling in the equations. We propose to use a similar scheme to (2.30), except now we have the following force terms with fixed points specified differently

$$
\begin{aligned}
\big(F_1(\boldsymbol{u}^h,\boldsymbol{v}^h)\big)_k^{(l+1)} &= \beta\Big(\big(u_1^h\big)_k^{(l+1)} + \big(v_1^h\big)_k^{(l+1)}\Big) \\
&\quad - \big(\partial_{u_1}^h T^h(x_1+u_1^{(l)}, x_2+u_2^{(l)})\big)_k\Big(\big(T^h(x_1+u_1^{(l+1)}, x_2+u_2^{(l)})\big)_k - \big(R^h(x_1,x_2)\big)_k\Big), \\
\big(F_2(\boldsymbol{u}^h,\boldsymbol{v}^h)\big)_k^{(l+1)} &= \beta\Big(\big(u_2^h\big)_k^{(l+1)} + \big(v_2^h\big)_k^{(l+1)}\Big) \\
&\quad - \big(\partial_{u_2}^h T^h(x_1+u_1^{(l)}, x_2+u_2^{(l)})\big)_k\Big(\big(T^h(x_1+u_1^{(l)}, x_2+u_2^{(l+1)})\big)_k - \big(R^h(x_1,x_2)\big)_k\Big),
\end{aligned}
$$

$$
\begin{aligned}
\big(G_1(\boldsymbol{u}^h,\boldsymbol{v}^h)\big)_k^{(l+1)} &= \beta\Big(\big(v_1^h\big)_k^{(l+1)} + \big(u_1^h\big)_k^{(l+1)}\Big) \\
&\quad - \big(\partial_{v_1}^h R^h(x_1+v_1^{(l)}, x_2+v_2^{(l)})\big)_k\Big(\big(R^h(x_1+v_1^{(l+1)}, x_2+v_2^{(l)})\big)_k - \big(T^h(x_1,x_2)\big)_k\Big), \\
\big(G_2(\boldsymbol{u}^h,\boldsymbol{v}^h)\big)_k^{(l+1)} &= \beta\Big(\big(v_2^h\big)_k^{(l+1)} + \big(u_2^h\big)_k^{(l+1)}\Big) \\
&\quad - \big(\partial_{v_2}^h R^h(x_1+v_1^{(l)}, x_2+v_2^{(l)})\big)_k\Big(\big(R^h(x_1+v_1^{(l)}, x_2+v_2^{(l+1)})\big)_k - \big(T^h(x_1,x_2)\big)_k\Big).
\end{aligned}
\tag{2.35}
$$

Again, after using Taylor approximations to linearise (2.35), at iteration step (l) we have the following smoother scheme which we use to compute the $(l+1)$ updates

$$
\begin{cases}
-\alpha\big(\Delta^h u_m^h\big)_k^{(l+1)} + \beta\Big(\big(u_m^h\big)_k^{(l+1)} + \big(v_m^h\big)_k^{(l+1)}\Big) \\
\qquad +\big(\partial_{u_m}^h T_{\boldsymbol{u}}^h\big)_k^{(l)}\Big[\big(T_{\boldsymbol{u}}^h\big)_k^{(l)} + \Big(\big(u_m^h\big)_k^{(l+1)} - \big(u_m^h\big)_k^{(l)}\Big)\big(\partial_{u_m}^h T_{\boldsymbol{u}}^h\big)_k^{(l)} - \big(R^h\big)_k\Big] = 0, \\
-\alpha\big(\Delta^h v_m^h\big)_k^{(l+1)}\beta\Big(\big(v_m^h\big)_k^{(l+1)} + \big(u_m^h\big)_k^{(l+1)}\Big) \\
\qquad +\big(\partial_{v_m}^h R_{\boldsymbol{v}}^h\big)_k^{(l)}\Big[\big(R_{\boldsymbol{v}}^h\big)_k^{(l)} + \Big(\big(v_m^h\big)_k^{(l+1)} - \big(v_m^h\big)_k^{(l)}\Big)\big(\partial_{v_m}^h R_{\boldsymbol{v}}^h\big)_k^{(l)} - \big(T^h\big)_k\Big] = 0
\end{cases}
\tag{2.36}
$$

for $m = 1, 2$. As we did for **S1** and **S2**, we use a scheme based upon a GSLEX method to compute the $(l+1)$ updates in (2.36).

3 Analysis for the NMG algorithm

As we mentioned at the end of §2.3, the effectiveness of the smoother scheme has a severe impact on the convergence of the NMG scheme. In order to determine how effective a given smoother scheme is within the NMG framework, we look to compute the so called 'smoothing rate' of the scheme which gives us an insight into how effectively the chosen smoother removes high frequency error components. However, before we look at computing the smoothing rates of our three proposed smoothers from §2.4, we must first determine whether each of the proposed smoothers are suitable for use as pointwise error smoothing procedures. To do this we must compute the h-ellipticity for each of the proposed smoothers. For both calculations (i.e. smoothing rates and h-ellipticity values) we can use local Fourier analysis or LFA.

3.1 Local Fourier Analysis (LFA)

In order to analyse the h-ellipticity and smoothing rate of a given smoother scheme, we can use a technique called LFA. Originally LFA was designed to only analyse the smoothing properties of discrete linear operators, however the work done by A. Brandt [4] proposed to locally 'freeze' the coefficients of non-linear operators thus enabling the use of LFA for non-linear operators such as the one in (2.30). In LFA [13,19], we begin by considering our problem over an infinite grid (thus removing any influence from the boundary conditions), and then assuming that the discrete form of a variable non-linear operator can be replaced by a constant linear operator and extended to this infinite grid, which we define as followed

$$
\Omega_h^\infty := \Big\{\boldsymbol{x} \in \Omega \colon \boldsymbol{x} = (x_1, x_2)^T = (ih, jh)^T \ \text{ for } i, j \in \mathbb{Z}^+\Big\}
\tag{3.1}
$$

with grid interval h defined by $h = \frac{1}{n-1}$. In addition let us also define the grid functions $\boldsymbol{\Phi}^h(\boldsymbol{x}, \boldsymbol{\theta}) = \exp\left(\frac{\boldsymbol{i\theta x}}{h}\right)$, where $\boldsymbol{\theta} = (\theta_1, \theta_2)^T \in \boldsymbol{\Theta} = [-\pi, \pi)^2$, $\boldsymbol{x} \in \Omega_h^\infty$ and $\boldsymbol{i} = \sqrt{-1}$, which when a discrete linear operator $\boldsymbol{\mathcal{L}}^h$ is applied gives

$$\boldsymbol{\mathcal{L}}^h \boldsymbol{\Phi}^h(\boldsymbol{x}, \boldsymbol{\theta}) = \hat{\boldsymbol{\mathcal{L}}}^h(\boldsymbol{\theta}) \boldsymbol{\Phi}^h(\boldsymbol{x}, \boldsymbol{\theta}) \tag{3.2}$$

where $\hat{\boldsymbol{\mathcal{L}}}^h(\boldsymbol{\theta})$ denotes the Fourier symbol of $\boldsymbol{\mathcal{L}}^h$ (see [45, 46]).

3.2 H-ellipticity measure for the proposed smoothers

For effective smoother schemes, the measure of the h-ellipticity is a key component which must be considered. This measure is used to ascertain whether a given smoother scheme, such as the ones we outlined in §2.4, are sufficient for use as pointwise error smoothing procedures for the given discrete operator within a multigrid framework; if not, one must consider line or block smoothers or problem reformulation.

We will now demonstrate that our proposed smoothers from §2.4 can be constructed for the given discrete operator, and can therefore be used in our proposed NMG scheme. To do this we use a similar calculation to the ones shown in [19, 30, 35, 45, 46] applied to the smoother schemes (2.32), (2.34) and (2.36) at some given outer iteration step.

H-Ellipticity for Smoother S1: We begin by writing (2.32) in the following operator form

$$\boldsymbol{\mathcal{L}}_1^h \boldsymbol{w}^h = \boldsymbol{\mathcal{G}}^h \tag{3.3}$$

with

$$\boldsymbol{\mathcal{L}}_1^h = \begin{pmatrix} -\alpha\Delta^h + \sigma_{11}^h + \beta & 0 & 0 & 0 \\ 0 & -\alpha\Delta^h + \sigma_{22}^h + \beta & 0 & 0 \\ 0 & 0 & -\alpha\Delta^h + \tau_{11}^h + \beta & 0 \\ 0 & 0 & 0 & -\alpha\Delta^h + \tau_{22}^h + \beta \end{pmatrix},$$

$$\boldsymbol{\mathcal{G}}^h = \begin{pmatrix} \boldsymbol{g}_1^h - F_1(\boldsymbol{u}^h, \boldsymbol{v}^h) \\ \boldsymbol{g}_2^h - F_2(\boldsymbol{u}^h, \boldsymbol{v}^h) \\ \boldsymbol{g}_3^h - G_1(\boldsymbol{u}^h, \boldsymbol{v}^h) \\ \boldsymbol{g}_4^h - G_2(\boldsymbol{u}^h, \boldsymbol{v}^h) \end{pmatrix}, \quad \boldsymbol{w}^h = \begin{pmatrix} \boldsymbol{u}_1^h \\ \boldsymbol{u}_2^h \\ \boldsymbol{v}_1^h \\ \boldsymbol{v}_2^h \end{pmatrix} \tag{3.4}$$

where

$$\begin{aligned} F_m(\boldsymbol{u}^h, \boldsymbol{v}^h) &= \left(\partial_{u_m}^h T_{\boldsymbol{u}}^h\right)^2 u_m^h - \beta v_m^h - \left(\partial_{u_m}^h T_{\boldsymbol{u}}^h\right)\left(T_{\boldsymbol{u}}^h - R^h\right), \\ G_m(\boldsymbol{u}^h, \boldsymbol{v}^h) &= \left(\partial_{v_m}^h R_{\boldsymbol{v}}^h\right)^2 v_m^h - \beta u_m^h - \left(\partial_{v_m}^h R_{\boldsymbol{v}}^h\right)\left(R_{\boldsymbol{v}}^h - T^h\right), \\ \sigma_{pq}^h &= \partial_{u_p}^h T_{\boldsymbol{u}}^h \partial_{u_q}^h T_{\boldsymbol{u}}^h, \; \tau_{pq}^h = \partial_{v_p}^h R_{\boldsymbol{v}}^h \partial_{v_q}^h R_{\boldsymbol{v}}^h \end{aligned} \tag{3.5}$$

for $m, p, q = 1, 2$. Since LFA is a local method for a nonlinear problem, we apply the analysis separately to each individual grid point. This then leads to a local discrete system which is only defined within a small neighbourhood of the discrete grid point (i, j). Applying our discrete linear operator $\boldsymbol{\mathcal{L}}_1^h$ to the grid functions $\boldsymbol{\Phi}^h(\boldsymbol{x}, \boldsymbol{\theta})$ yields the following

$$\boldsymbol{\mathcal{L}}_1^h \boldsymbol{\Phi}^h(\boldsymbol{x}, \boldsymbol{\theta}) = \hat{\boldsymbol{\mathcal{L}}}_1^h(\boldsymbol{\theta}) \boldsymbol{\Phi}^h(\boldsymbol{x}, \boldsymbol{\theta}) \tag{3.6}$$

where $\hat{\boldsymbol{\mathcal{L}}}_1^h(\boldsymbol{\theta})$ denotes the Fourier symbol of the operator $\boldsymbol{\mathcal{L}}_1^h$, and is given by (letting $a = \beta - \alpha\hat{\mathscr{L}}^h(\boldsymbol{\theta})$)

$$\hat{\boldsymbol{\mathcal{L}}}_1^h(\boldsymbol{\theta}) = \begin{pmatrix} \sigma_{11}^h + a & 0 & 0 & 0 \\ 0 & \sigma_{22}^h + a & 0 & 0 \\ 0 & 0 & \tau_{11}^h + a & 0 \\ 0 & 0 & 0 & \tau_{22}^h + a \end{pmatrix} \tag{3.7}$$

also with $\hat{\mathscr{L}}^h(\boldsymbol{\theta})$ denoting the Fourier symbol of the discrete Laplace operator Δ^h. Then, the h-ellipticity measure is calculated from the following

$$\mathscr{E}_1^h\left(\boldsymbol{\mathcal{L}}_1^h\right) = \frac{\min\left\{\left|\det\left(\hat{\boldsymbol{\mathcal{L}}}_1^h(\boldsymbol{\theta})\right)\right| : \boldsymbol{\theta}\in\boldsymbol{\Theta}_{high}\right\}}{\max\left\{\left|\det\left(\hat{\boldsymbol{\mathcal{L}}}_1^h(\boldsymbol{\theta})\right)\right| : \boldsymbol{\theta}\in\boldsymbol{\Theta}\right\}} \tag{3.8}$$

where $\boldsymbol{\Theta} = [-\pi,\pi)^2$ and $\boldsymbol{\Theta}_{high} = \boldsymbol{\Theta}\backslash\left[-\frac{\pi}{2},\frac{\pi}{2}\right)^2$ denotes the high frequency range. It can be shown that

$$\begin{aligned}\det\left(\hat{\boldsymbol{\mathcal{L}}}^h(\boldsymbol{\theta})\right) &= \alpha^4\left(\hat{\mathscr{L}}^h(\boldsymbol{\theta})\right)^4 - \alpha^3(d_1+c_1)\left(\hat{\mathscr{L}}^h(\boldsymbol{\theta})\right)^3 + \alpha^2(d_2+c_1d_1+c_2)\left(\hat{\mathscr{L}}^h(\boldsymbol{\theta})\right)^2\\ &\quad - \alpha(c_1d_2+c_2d_1)\left(\hat{\mathscr{L}}^h(\boldsymbol{\theta})\right) + c_2d_2\end{aligned} \tag{3.9}$$

where

$$\begin{aligned}&c_1 = \sigma_{11}^h + \sigma_{22}^h + 2\beta,\ c_2 = \sigma_{11}^h\sigma_{22}^h + \beta\left(\sigma_{11}^h + \sigma_{22}^h\right) + \beta^2\\ &d_1 = \tau_{11}^h + \tau_{22}^h + 2\beta,\ d_2 = \tau_{11}^h\tau_{22}^h + \beta\left(\tau_{11}^h + \tau_{22}^h\right) + \beta^2.\end{aligned} \tag{3.10}$$

From [19], it was shown that

$$-\hat{\mathscr{L}}^h(\boldsymbol{\theta}) = \frac{2}{h^2}\Big(2 - (\cos\theta_1 + \cos\theta_2)\Big), \tag{3.11}$$

$$\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}_{high}}\left(-\hat{\mathscr{L}}^h(\boldsymbol{\theta})\right) = \frac{2}{h^2},\ \max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}}\left(-\hat{\mathscr{L}}^h(\boldsymbol{\theta})\right) = \frac{8}{h^2} \tag{3.12}$$

thus (3.8) becomes

$$\begin{aligned}\mathscr{E}_1^h\left(\hat{\boldsymbol{\mathcal{L}}}_1^h(\boldsymbol{\theta})\right) &= \frac{\left(\frac{16\alpha^4}{h^8} + \frac{8\alpha^3(d_1+c_1)}{h^6} + \frac{4\alpha^2(d_+c_1d_1+c_2)}{h^4} + \frac{2\alpha(c_1d_2+c_2d_1)}{h^2} + c_2d_2\right)}{\left(\frac{4096\alpha^4}{h^8} + \frac{512\alpha^3(d_1+c_1)}{h^6} + \frac{64\alpha^2(d_+c_1d_1+c_2)}{h^4} + \frac{8\alpha(c_1d_2+c_2d_1)}{h^2} + c_2d_2\right)}\\ &= \frac{\left(\begin{aligned}16\alpha^4 + 8\alpha^3(d_1+c_1)h^2 + 4\alpha^2(d_+c_1d_1+c_2)h^4&\\ + 2\alpha(c_1d_2+c_2d_1)h^6 + c_2d_2h^8&\end{aligned}\right)}{\left(\begin{aligned}4096\alpha^4 + 512\alpha^3(d_1+c_1)h^2 + 64\alpha^2(d_+c_1d_1+c_2)h^4&\\ + 8\alpha(c_1d_2+c_2d_1)h^6 + (c_2d_2)h^8&\end{aligned}\right)}\end{aligned} \tag{3.13}$$

and so, taking the limit as $h\to 0$, we get

$$\lim_{h\to 0}\mathscr{E}_1^h\left(\hat{\boldsymbol{\mathcal{L}}}_1^h(\boldsymbol{\theta})\right) = \frac{1}{256}. \tag{3.14}$$

From this result, we can conclude that the h-ellipticity measure is always bounded away from 0 regardless of the values of α, β, h, σ_{pq}^h, τ_{pq}^h for $p,q = 1,2$. Or in other words, the results do not depend on the given images R, T, the choice of parameters α, β or the mesh interval h. Therefore we can conclude that smoother **S1** is sufficient for use as a pointwise error smoothing procedure.

H-Ellipticity for Smoother S2: Now we repeat the h-ellipticity calculation procedure for smoother **S2**. Similar to smoother **S1**, we get the following Fourier symbol for the operator $\boldsymbol{\mathcal{L}}_2^h$ (again $a = \beta - \alpha\hat{\mathscr{L}}^h(\boldsymbol{\theta})$)

$$\hat{\boldsymbol{\mathcal{L}}}_2^h(\boldsymbol{\theta}) = \begin{pmatrix}\sigma_{11}^h + a & \sigma_{12}^h & \beta & 0\\ \sigma_{12}^h & \sigma_{22}^h + a & 0 & \beta\\ \beta & 0 & \tau_{11}^h + a & \tau_{12}^h\\ 0 & \beta & \tau_{12}^h & \tau_{22}^h + a\end{pmatrix} \tag{3.15}$$

where $\mathscr{L}^h(\boldsymbol{\theta})$ again denotes the Fourier symbol of Δ^h and $\sigma_{pq}^h, \tau_{pq}^h$ are as in (3.5). The h-ellipticity for $\boldsymbol{\mathcal{L}}_2^h$ is computed using

$$\mathscr{E}_2^h\left(\boldsymbol{\mathcal{L}}_2^h\right) = \frac{\min\left\{\left|\det\left(\hat{\boldsymbol{\mathcal{L}}}_2^h(\boldsymbol{\theta})\right)\right| : \boldsymbol{\theta}\in\boldsymbol{\Theta}_{high}\right\}}{\max\left\{\left|\det\left(\hat{\boldsymbol{\mathcal{L}}}_2^h(\boldsymbol{\theta})\right)\right| : \boldsymbol{\theta}\in\boldsymbol{\Theta}\right\}}. \tag{3.16}$$

Simplifying the determinant we get

$$
\begin{aligned}
\det\left(\hat{\boldsymbol{\mathcal{L}}}_2^h(\boldsymbol{\theta})\right) &= \left(\sigma_{11}^h + a\right)\left(\sigma_{22}^h + a\right)\left(\tau_{11}^h + a\right)\left(\tau_{22}^h + a\right) - \left(\sigma_{11}^h + a\right)\left(\sigma_{22}^h + a\right)\left(\tau_{12}^h\right)^2 \\
&\quad - \left(\tau_{11}^h + a\right)\left(\tau_{22}^h + a\right)\left(\sigma_{12}^h\right)^2 - \left(\sigma_{11}^h + a\right)\left(\tau_{22}^h + a\right)\beta^2 \\
&\quad - \left(\sigma_{22}^h + a\right)\left(\tau_{22}^h + a\right)\beta^2 + \left(\sigma_{12}^h\right)^2\left(\tau_{12}^h\right)^2 - 2\sigma_{12}^h\tau_{12}^h\beta^2 + \beta^4 \\
&= \alpha^4\left(\hat{\mathscr{L}}^h(\boldsymbol{\theta})\right)^4 - \alpha^3\left(d_1 + c_1\right)\left(\hat{\mathscr{L}}^h(\boldsymbol{\theta})\right)^3 \\
&\quad + \alpha^2\left(d_2 + c_1 d_1 + c_2 - c_5 - d_5 + 2\beta^2\right)\left(\hat{\mathscr{L}}^h(\boldsymbol{\theta})\right)^2 \\
&\quad - \alpha\left(c_1 d_2 + c_2 d_1 + c_3 + d_3 + c_1 d_5 + d_1 c_5\right)\left(\hat{\mathscr{L}}^h(\boldsymbol{\theta})\right) \\
&\quad + c_2 d_2 + c_4 + d_5 - d_2 c_5 - c_2 d_5 + c_5 d_5 + 2\beta^4
\end{aligned} \tag{3.17}
$$

where c_1, c_2, d_1, d_2 are as in (3.10), and

$$
\begin{aligned}
c_3 &= \beta^2\left(\sigma_{11}^h + \tau_{11}^h + 2\beta\right),\ c_4 = \beta^2\left(\beta^2 + \beta\left(\sigma_{11}^h + \tau_{11}^h\right) + \sigma_{11}^h + \tau_{11}^h\right),\ c_5 = \left(\sigma_{12}^h\right)^2 \\
d_3 &= \beta^2\left(\sigma_{22}^h + \tau_{22}^h + 2\beta\right),\ d_4 = \beta^2\left(\beta^2 + \beta\left(\sigma_{22}^h + \tau_{22}^h\right) + \sigma_{22}^h + \tau_{22}^h\right),\ d_5 = \left(\tau_{12}^h\right)^2.
\end{aligned} \tag{3.18}
$$

From the h-ellipticity calculation of smoother **S1**, we see that the value of the limit (3.14) as $h \to 0$ depends only on the coefficient of the α^4 term. Thus we get

$$
\lim_{h \to 0} \mathscr{E}_2^h\left(\hat{\boldsymbol{\mathcal{L}}}_2^h(\boldsymbol{\theta})\right) = \frac{1}{256} \tag{3.19}
$$

and so smoother **S2** is suitable for use as a pointwise error smoothing procedure.

<u>H-Ellipticity for Smoother S3</u>: Finally we once again repeat the h-ellipticity calculation for our simplified smoother **S3**. Doing so gives the following Fourier symbol for the operator $\boldsymbol{\mathcal{L}}_3^h$

$$
\hat{\boldsymbol{\mathcal{L}}}_3^h(\boldsymbol{\theta}) = \begin{pmatrix} \sigma_{11}^h + a & 0 & \beta & 0 \\ 0 & \sigma_{22}^h + a & 0 & \beta \\ \beta & 0 & \tau_{11}^h + a & 0 \\ 0 & \beta & 0 & \tau_{22}^h + a \end{pmatrix} \tag{3.20}
$$

where $\hat{\mathscr{L}}^h(\boldsymbol{\theta})$ again denotes the Fourier symbol of the discrete Laplace operator Δ^h and $\sigma_{pq}^h, \tau_{pq}^h$ are as defined in (3.5) for $p, q = 1, 2$. We compute the h-ellipticity using the following

$$
\mathscr{E}_3^h\left(\boldsymbol{\mathcal{L}}_3^h\right) = \frac{\min\left\{\left|\det\left(\hat{\boldsymbol{\mathcal{L}}}_3^h(\boldsymbol{\theta})\right)\right| : \boldsymbol{\theta} \in \boldsymbol{\Theta}_{high}\right\}}{\max\left\{\left|\det\left(\hat{\boldsymbol{\mathcal{L}}}_3^h(\boldsymbol{\theta})\right)\right| : \boldsymbol{\theta} \in \boldsymbol{\Theta}\right\}}. \tag{3.21}
$$

Further from

$$
\begin{aligned}
\det\left(\hat{\boldsymbol{\mathcal{L}}}_3^h(\boldsymbol{\theta})\right) &= \left(\sigma_{11}^h + a\right)\left(\sigma_{22}^h + a\right)\left(\tau_{11}^h + a\right)\left(\tau_{22}^h + a\right) \\
&\quad - \left(\sigma_{11}^h + a\right)\left(\tau_{11}^h + a\right)\beta^2 - \left(\sigma_{22}^h + a\right)\left(\tau_{22}^h + a\right)\beta^2 + \beta^4 \\
&= \alpha^4\left(\hat{\mathscr{L}}^h(\boldsymbol{\theta})\right)^4 - \alpha^3\left(d_1 + c_1\right)\left(\hat{\mathscr{L}}^h(\boldsymbol{\theta})\right)^3 + \alpha^2\left(d_2 + c_1 d_1 + c_2 + 2\beta^2\right)\left(\hat{\mathscr{L}}^h(\boldsymbol{\theta})\right)^2 \\
&\quad - \alpha\left(c_1 d_2 + c_2 d_1 + c_3 + d_3\right)\left(\hat{\mathscr{L}}^h(\boldsymbol{\theta})\right) + c_2 d_2 + c_4 + d_4 + \beta^4
\end{aligned} \tag{3.22}
$$

where c_1, c_2, d_1, d_2 are as given in (3.10) and c_3, c_4, d_3, d_4 are as given in (3.18), we get the following

$$
\lim_{h \to 0} \mathscr{E}_3^h\left(\hat{\boldsymbol{\mathcal{L}}}_3^h(\boldsymbol{\theta})\right) = \frac{1}{256}. \tag{3.23}
$$

Thus we reach the same conclusion, namely the h-ellipticity is always bounded away from 0, and so smoother **S3** is sufficient for use as a pointwise error smoothing procedure.

3.3 Smoother analysis of the proposed smoothers

We now consider how effective our smoother schemes from §2.4 are at removing high frequency error components. The discrete residual error, as shown in §2.3, can be split into the sum of low frequency error components (which can be well approximated on a coarser grid) and high frequency error components (which disappear on coarser grids due to aliasing). For this reason, one key aspect of the NMG framework is the removal of all high frequency error components before we restrict to a coarser grid. We can use LFA to approximate the smoothing rate of a given smoother scheme, and from this we can obtain an estimate of how many smoothing steps we will need to remove the high frequency components if we aim to reduce the error by 10^{-1} (typical in a NMG context).

LFA for Smoother S1: We begin our calculation of the smoothing rate by writing the discrete system (2.32) in the following way

$$\boldsymbol{\mathcal{L}}_1^h \boldsymbol{w}^h + \boldsymbol{\mathcal{M}}_1^h \boldsymbol{w}^h = \boldsymbol{\mathcal{G}}^h \tag{3.24}$$

where $\boldsymbol{\mathcal{L}}_1^h, \boldsymbol{w}^h, \boldsymbol{\mathcal{G}}^h$ are as defined in (3.4), and

$$\boldsymbol{\mathcal{M}}_1^h = \begin{pmatrix} -\sigma_{11}^h & 0 & \beta & 0 \\ 0 & -\sigma_{22}^h & 0 & \beta \\ \beta & 0 & -\tau_{11}^h & 0 \\ 0 & \beta & 0 & -\tau_{22}^h \end{pmatrix} \tag{3.25}$$

with $\sigma_{pq}^h, \tau_{pq}^h$ as in (3.5) for $p, q = 1, 2$. Also we can rewrite the discrete Laplace operator as $\Delta^h = \mathscr{L}_+^h + \mathscr{L}_0^h + \mathscr{L}_-^h$, where $\mathscr{L}_+^h$, $\mathscr{L}_0^h$, $\mathscr{L}_-^h$ define the following stencils

$$\mathscr{L}_+^h = \frac{1}{h^2}\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \mathscr{L}_0^h = \frac{1}{h^2}\begin{bmatrix} 0 & 0 & 0 \\ 0 & -4 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \mathscr{L}_-^h = \frac{1}{h^2}\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.26}$$

and so, we can write (3.24) in the following way

$$\boldsymbol{\mathcal{L}}_{1+}^h \boldsymbol{u}_{new}^h + \boldsymbol{\mathcal{L}}_{1\,0}^h \boldsymbol{u}_{new}^h + \boldsymbol{\mathcal{L}}_{1-}^h \boldsymbol{u}_{old}^h + \boldsymbol{\mathcal{M}}_1^h \boldsymbol{u}_{old}^h = \boldsymbol{\mathcal{G}}^h \tag{3.27}$$

where we have denoted the current and previous approximations of $\boldsymbol{u}^h, \boldsymbol{v}^h$ by $\boldsymbol{u}_{new}^h, \boldsymbol{v}_{new}^h$ and $\boldsymbol{u}_{old}^h, \boldsymbol{v}_{old}^h$ respectively, also with

$$\boldsymbol{\mathcal{L}}_{1+}^h = \begin{pmatrix} -\alpha\mathscr{L}_+^h & 0 & 0 & 0 \\ 0 & -\alpha\mathscr{L}_+^h & 0 & 0 \\ 0 & 0 & -\alpha\mathscr{L}_+^h & 0 \\ 0 & 0 & 0 & -\alpha\mathscr{L}_+^h \end{pmatrix}, \boldsymbol{\mathcal{L}}_{1-}^h = \begin{pmatrix} -\alpha\mathscr{L}_-^h & 0 & 0 & 0 \\ 0 & -\alpha\mathscr{L}_-^h & 0 & 0 \\ 0 & 0 & -\alpha\mathscr{L}_-^h & 0 \\ 0 & 0 & 0 & -\alpha\mathscr{L}_-^h \end{pmatrix}$$

$$\boldsymbol{\mathcal{L}}_{1\,0}^h = \begin{pmatrix} -\alpha\mathscr{L}_0^h + \sigma_{11}^h + \beta & 0 & 0 & 0 \\ 0 & -\alpha\mathscr{L}_0^h + \sigma_{22}^h + \beta & 0 & 0 \\ 0 & 0 & -\alpha\mathscr{L}_0^h + \tau_{11}^h + \beta & 0 \\ 0 & 0 & 0 & -\alpha\mathscr{L}_0^h + \tau_{22}^h + \beta \end{pmatrix}$$

$$\boldsymbol{\mathcal{M}}_1^h = \begin{pmatrix} -\sigma_{11}^h & 0 & \beta & 0 \\ 0 & -\sigma_{22}^h & 0 & \beta \\ \beta & 0 & -\tau_{11}^h & 0 \\ 0 & \beta & 0 & -\tau_{22}^h \end{pmatrix}. \tag{3.28}$$

Now subtracting (3.27) from (3.24) we can obtain the local error equations given by

$$\left[\boldsymbol{\mathcal{L}}_{1+}^h + \boldsymbol{\mathcal{L}}_{1\,0}^h\right] \boldsymbol{e}_{new}^h = -\left[\boldsymbol{\mathcal{L}}_{1-}^h + \boldsymbol{\mathcal{M}}_1^h\right] \boldsymbol{e}_{old}^h \tag{3.29}$$

where $\boldsymbol{e}_*^h = \left(e_{1\,*}^h, e_{2\,*}^h, e_{3\,*}^h, e_{4\,*}^h\right)^T$. Then we expand the local errors in (3.29) using Fourier components to give

$$\boldsymbol{e}_{new}^h = \sum_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \boldsymbol{\psi}_{\boldsymbol{\theta}}^{new} \exp\left(\frac{2\boldsymbol{i}\theta_1 i\pi}{h} + \frac{2\boldsymbol{i}\theta_2 j\pi}{h}\right), \boldsymbol{e}_{old}^h = \sum_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \boldsymbol{\psi}_{\boldsymbol{\theta}}^{old} \exp\left(\frac{2\boldsymbol{i}\theta_1 i\pi}{h} + \frac{2\boldsymbol{i}\theta_2 j\pi}{h}\right) \tag{3.30}$$

where $\boldsymbol{\psi}_{\boldsymbol{\theta}}^{*}$ are Fourier coefficients, $\boldsymbol{i}=\sqrt{-1}$ and $\boldsymbol{\Theta}=[-\pi,\pi)^2$. Using the Fourier component form of the errors in (3.30), allows us to rewrite the local error equation (3.29) in terms of these Fourier components. Then we get

$$\left[\hat{\mathcal{L}}_{1+}^{h}(\boldsymbol{\theta})+\hat{\mathcal{L}}_{1\,0}^{h}(\boldsymbol{\theta})\right]\boldsymbol{\psi}_{\boldsymbol{\theta}}^{new}\exp\left(\frac{2\boldsymbol{i}\theta_1 i\pi}{h}+\frac{2\boldsymbol{i}\theta_2 j\pi}{h}\right)=-\left[\hat{\mathcal{L}}_{1-}^{h}(\boldsymbol{\theta})+\hat{\mathcal{M}}_{1}^{h}(\boldsymbol{\theta})\right]\boldsymbol{\psi}_{\boldsymbol{\theta}}^{old}\exp\left(\frac{2\boldsymbol{i}\theta_1 i\pi}{h}+\frac{2\boldsymbol{i}\theta_2 j\pi}{h}\right) \tag{3.31}$$

where

$$\begin{aligned}
\hat{\mathcal{L}}_{1+}^{h}(\boldsymbol{\theta})&=\begin{pmatrix}-\frac{\alpha}{h^2}\left(e^{-\boldsymbol{i}\omega_2}+e^{-\boldsymbol{i}\omega_1}\right)&0&0&0\\0&-\frac{\alpha}{h^2}\left(e^{-\boldsymbol{i}\omega_2}+e^{-\boldsymbol{i}\omega_1}\right)&0&0\\0&0&-\frac{\alpha}{h^2}\left(e^{-\boldsymbol{i}\omega_2}+e^{-\boldsymbol{i}\omega_1}\right)&0\\0&0&0&-\frac{\alpha}{h^2}\left(e^{-\boldsymbol{i}\omega_2}+e^{-\boldsymbol{i}\omega_1}\right)\end{pmatrix}\\
\hat{\mathcal{L}}_{1-}^{h}(\boldsymbol{\theta})&=\begin{pmatrix}-\frac{\alpha}{h^2}\left(e^{\boldsymbol{i}\omega_2}+e^{\boldsymbol{i}\omega_1}\right)&0&0&0\\0&-\frac{\alpha}{h^2}\left(e^{\boldsymbol{i}\omega_2}+e^{\boldsymbol{i}\omega_1}\right)&0&0\\0&0&-\frac{\alpha}{h^2}\left(e^{\boldsymbol{i}\omega_2}+e^{\boldsymbol{i}\omega_1}\right)&0\\0&0&0&-\frac{\alpha}{h^2}\left(e^{\boldsymbol{i}\omega_2}+e^{\boldsymbol{i}\omega_1}\right)\end{pmatrix}\\
\hat{\mathcal{L}}_{1\,0}^{h}(\boldsymbol{\theta})&=\begin{pmatrix}\frac{4\alpha}{h^2}+\sigma_{11}^h+\beta&0&0&0\\0&\frac{4\alpha}{h^2}+\sigma_{22}^h+\beta&0&0\\0&0&\frac{4\alpha}{h^2}+\tau_{11}^h+\beta&0\\0&0&0&\frac{4\alpha}{h^2}+\tau_{22}^h+\beta\end{pmatrix}\\
\hat{\mathcal{M}}_{1}^{h}(\boldsymbol{\theta})&=\begin{pmatrix}-\sigma_{11}^h&0&\beta&0\\0&-\sigma_{22}^h&0&\beta\\\beta&0&-\tau_{11}^h&0\\0&\beta&0&-\tau_{22}^h\end{pmatrix}
\end{aligned}\tag{3.32}$$

and with $\omega_m=\frac{2\pi\theta_m}{h}$ for $m=1,2$. Finally, we compute the local smoothing rate using the following

$$\mu_{loc}\equiv\mu_{loc}(\boldsymbol{\theta})=\sup\left\{\rho\left(\hat{\mathcal{S}}_{1}^{h}(\boldsymbol{\theta})\right):\boldsymbol{\theta}\in\boldsymbol{\Theta}_{high}\right\}\tag{3.33}$$

where $\boldsymbol{\Theta}_{high}=[-\pi,\pi)^2\setminus\left[-\frac{\pi}{2},\frac{\pi}{2}\right)^2$ denotes the high frequency range, $\rho(\cdot)$ denotes the spectral radius and $\hat{\mathcal{S}}_{1}^{h}(\boldsymbol{\theta})$ denotes the amplification matrix given by the following

$$\hat{\mathcal{S}}_{1}^{h}(\boldsymbol{\theta})=-\left[\hat{\mathcal{L}}_{1+}^{h}(\boldsymbol{\theta})+\hat{\mathcal{L}}_{1\,0}^{h}(\boldsymbol{\theta})\right]^{-1}\left[\hat{\mathcal{L}}_{1-}^{h}(\boldsymbol{\theta})+\hat{\mathcal{M}}_{1}^{h}(\boldsymbol{\theta})\right]\tag{3.34}$$

for $m=1,2$.

<u>LFA for Smoother S2</u>: Now we repeat the smoothing rate calculation we used for smoother **S1**, but this time for smoother **S2**. Doing so we compute the local smoothing rate from

$$\mu_{loc}\equiv\mu_{loc}(\boldsymbol{\theta})=\sup\left\{\rho\left(\hat{\mathcal{S}}_{2}^{h}(\boldsymbol{\theta})\right):\boldsymbol{\theta}\in\boldsymbol{\Theta}_{high}\right\}\tag{3.35}$$

with amplification matrix

$$\hat{\mathcal{S}}_{2}^{h}(\boldsymbol{\theta})=-\left[\hat{\mathcal{L}}_{2+}^{h}(\boldsymbol{\theta})+\hat{\mathcal{L}}_{2\,0}^{h}(\boldsymbol{\theta})\right]^{-1}\left[\hat{\mathcal{L}}_{2-}^{h}(\boldsymbol{\theta})+\hat{\mathcal{M}}_{2}^{h}(\boldsymbol{\theta})\right]\tag{3.36}$$

where $\hat{\mathcal{L}}_{2+}^{h}(\boldsymbol{\theta})$ and $\hat{\mathcal{L}}_{2-}^{h}(\boldsymbol{\theta})$ are the same as $\hat{\mathcal{L}}_{1+}^{h}(\boldsymbol{\theta})$ and $\hat{\mathcal{L}}_{1-}^{h}(\boldsymbol{\theta})$ from (3.32) respectively, and

$$\begin{aligned}
\hat{\mathcal{L}}_{2\,0}^{h}(\boldsymbol{\theta})&=\begin{pmatrix}\frac{4\alpha}{h^2}+\sigma_{11}^h+\beta&\sigma_{12}^h&\beta&0\\\sigma_{12}^h&\frac{4\alpha}{h^2}+\sigma_{22}^h+\beta&0&\beta\\\beta&0&\frac{4\alpha}{h^2}+\tau_{11}^h+\beta&\tau_{12}^h\\0&\beta&\tau_{12}^h&\frac{4\alpha}{h^2}+\tau_{22}^h+\beta\end{pmatrix}\\
\hat{\mathcal{M}}_{2}^{h}(\boldsymbol{\theta})&=\begin{pmatrix}-\sigma_{11}^h&-\sigma_{12}^h&0&0\\-\sigma_{12}^h&-\sigma_{22}^h&0&0\\0&0&-\tau_{11}^h&-\tau_{12}^h\\0&0&-\tau_{12}^h&-\tau_{22}^h\end{pmatrix}
\end{aligned}\tag{3.37}$$

Remark 3.1. *We remark that if we set $\beta = 0$, then the smoother analysis becomes similar to that shown in [19]. However the analysis in [19] led to an overestimation of the smoothing rate due to omitting the lagged displacements (as shown by the $\hat{\boldsymbol{\mathcal{M}}}_2^h(\boldsymbol{\theta})$ matrix), which resulted in an underestimation of the number of smoother steps required and thus a less effective NMG scheme.*

LFA for Smoother S3: Again we repeat the smoothing rate calculation, this time for smoother **S3**. We compute the local smoothing rate using the following

$$\mu_{loc} \equiv \mu_{loc}(\boldsymbol{\theta}) = \sup\left\{\rho\left(\hat{\boldsymbol{\mathcal{S}}}_3^h(\boldsymbol{\theta})\right) : \boldsymbol{\theta} \in \boldsymbol{\Theta}_{high}\right\} \tag{3.38}$$

with amplification matrix

$$\hat{\boldsymbol{\mathcal{S}}}_3^h(\boldsymbol{\theta}) = -\left[\hat{\boldsymbol{\mathcal{L}}}_{3+}^h(\boldsymbol{\theta}) + \hat{\boldsymbol{\mathcal{L}}}_{3\,0}^h(\boldsymbol{\theta})\right]^{-1}\left[\hat{\boldsymbol{\mathcal{L}}}_{3-}^h(\boldsymbol{\theta}) + \hat{\boldsymbol{\mathcal{M}}}_3^h(\boldsymbol{\theta})\right] \tag{3.39}$$

where $\hat{\boldsymbol{\mathcal{L}}}_{3+}^h(\boldsymbol{\theta})$ and $\hat{\boldsymbol{\mathcal{L}}}_{3-}^h(\boldsymbol{\theta})$ are the same as $\hat{\boldsymbol{\mathcal{L}}}_{1+}^h(\boldsymbol{\theta})$ and $\hat{\boldsymbol{\mathcal{L}}}_{1-}^h(\boldsymbol{\theta})$ from (3.32) respectively, and

$$\begin{aligned}\hat{\boldsymbol{\mathcal{L}}}_{3\,0}^h(\boldsymbol{\theta}) &= \begin{pmatrix} \frac{4\alpha}{h^2} + \sigma_{11}^h + \beta & 0 & \beta & 0 \\ 0 & \frac{4\alpha}{h^2} + \sigma_{22}^h + \beta & 0 & \beta \\ \beta & 0 & \frac{4\alpha}{h^2} + \tau_{11}^h + \beta & 0 \\ 0 & \beta & 0 & \frac{4\alpha}{h^2} + \tau_{22}^h + \beta \end{pmatrix} \\ \hat{\boldsymbol{\mathcal{M}}}_3^h(\boldsymbol{\theta}) &= \begin{pmatrix} -\sigma_{11}^h & 0 & 0 & 0 \\ 0 & -\sigma_{22}^h & 0 & 0 \\ 0 & 0 & -\tau_{11}^h & 0 \\ 0 & 0 & 0 & -\tau_{22}^h \end{pmatrix}.\end{aligned} \tag{3.40}$$

Smoothing Rate Examples:

α	β	**S1**		**S2**		**S3**	
		μ_{avg}	Tol 10^{-1}	μ_{avg}	Tol 10^{-1}	μ_{avg}	Tol 10^{-1}
$\frac{1}{15}$	0	0.72942	8	0.73352	8	0.72942	8
	10^2	0.79205	10	0.72972	8	0.72526	8
	10^4	0.93335	34	0.73178	8	0.72545	8

Table 1: Comparison of the smoothing rates of the proposed smoothers **S1**-**S3** for parameters $\alpha = \frac{1}{15}$ and $\beta = 0, 10^2, 10^4$ after 5 inner and outer iterations on a 32×32 grid for Example 2 as shown in Figure 2. For each smoother, the smoothing rates and number of inner iterations required to reach an error reduction of 10^{-1} are shown.

From Table 1 we see that as the value of β increases the smoothing rate for smoother **S1** gets closer to 1. For this reason we conclude that smoother **S1** is not suitable for use in the NMG framework as this increase in smoothing rate would require an unreasonable number of smoother steps for practical applications as shown by the number of iterations required to reduce the error to a tolerance of 10^{-1} from Table 1. We also see that the rates for smoothers **S2** and **S3** remain stable even as the value of β increases. In addition, owing to this stability, we see that for both smoothers **S2** and **S3** 8 smoother steps are sufficient to reduce the error to a reasonable level before restriction.

3.4 Coarsest grid solvers

By using a NMG framework we are able to restrict our original problem on a large grid to a very coarse grid (e.g. 8×8). On this coarsest grid our aim is to solve the problem as accurately as possible, owing to the low computational cost, and so we need a designated solver for use only on this coarsest grid. Here we will present 2 coarsest grid solvers, based upon smoothers **S2** and **S3** from §2.4. It is also possible to estimate the convergence rate of a given coarse grid solver using (3.33) with $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ instead of only being restricted to the high frequency range $\boldsymbol{\Theta}_{high}$, and from this rate we can approximate the number of iterations required to reach a desired error tolerance similar to what we did with the smoothing rates. However this analysis can only be performed on a very coarse grid, such as a 8×8 grid, and in this paper we do not present the details of this analysis.

First Proposed Coarsest Level Solver C1: From §2.4, we know that on the coarsest grid we are looking to solve the system of equations shown in (2.34) with coarse grid interval width H instead of the fine grid interval width h. Equivalently we can express the system (2.34) in the following matrix form

$$\bar{\boldsymbol{A}}^H \boldsymbol{w}^H = \bar{\boldsymbol{F}}^H \tag{3.41}$$

where $\bar{\boldsymbol{A}}^H \in \mathbb{R}^{4(n-2)^2 \times 4(n-2)^2}$ and $\boldsymbol{w}^H, \bar{\boldsymbol{F}}^H \in \mathbb{R}^{4(n-2)^2 \times 1}$ are given by

$$\bar{\boldsymbol{A}}^H = \begin{pmatrix} \boldsymbol{A}_1^H & \tilde{\boldsymbol{A}}_1^H & \boldsymbol{I}_2 & \boldsymbol{0} \\ \tilde{\boldsymbol{A}}_2^H & \boldsymbol{A}_2^H & \boldsymbol{0} & \boldsymbol{I}_2 \\ \boldsymbol{I}_2 & \boldsymbol{0} & \boldsymbol{B}_1^H & \tilde{\boldsymbol{B}}_1^H \\ \boldsymbol{0} & \boldsymbol{I}_2 & \tilde{\boldsymbol{B}}_2^H & \boldsymbol{B}_2^H \end{pmatrix}, \ \boldsymbol{w} = \begin{pmatrix} \boldsymbol{u}_1^H \\ \boldsymbol{u}_2^H \\ \boldsymbol{v}_1^H \\ \boldsymbol{v}_2^H \end{pmatrix}, \ \bar{\boldsymbol{F}} = \begin{pmatrix} \bar{\boldsymbol{F}}_1^H \\ \bar{\boldsymbol{F}}_2^H \\ \bar{\boldsymbol{G}}_1^H \\ \bar{\boldsymbol{G}}_2^H \end{pmatrix} \tag{3.42}$$

where $\boldsymbol{A}_s^H, \boldsymbol{B}_s^H \in \mathbb{R}^{(n-2)^2 \times (n-2)^2}$ are the block tri-diagonal system matrices reflecting the coefficients of the $\left(u_s^H\right)_*^{(l+1)}, \left(v_s^H\right)_*^{(l+1)}$ terms at the various neighbouring pixels for each discrete interior point k respectively, $\tilde{\boldsymbol{A}}_s^H, \tilde{\boldsymbol{B}}_s^H \in \mathbb{R}^{(n-2)^2 \times (n-2)^2}$ are the diagonal matrices corresponding to the $\left(u_t^H\right)_*^{(l+1)}, \left(v_t^H\right)_*^{(l+1)}$ terms in the $\left(u_s^H\right)_k^{(l+1)}, \left(v_s^H\right)_k^{(l+1)}$ equations respectively, $\boldsymbol{I}_2 = \beta \boldsymbol{I}$ where $\boldsymbol{I}$ denotes the $(n-2)^2 \times (n-2)^2$ identity matrix and $\boldsymbol{u}_s^H, \boldsymbol{v}_s^H, \bar{\boldsymbol{F}}_s^H, \bar{\boldsymbol{G}}_s^H \in \mathbb{R}^{(n-2)^2 \times 1}$ are the column vectors consisting of the displacements $\left(u_s^H\right)_k^{(l+1)}, \left(v_s^H\right)_k^{(l+1)}$ and RHS terms $\left(\bar{F}_s^H\right)_k^{(l+1)}, \left(\bar{G}_s^H\right)_k^{(l+1)}$ given by

$$\begin{aligned} \left(\bar{F}_s^H\right)_k &= \left(\left(\partial_{u_s}^H T_{\boldsymbol{u}}^H\right)^2\right)_k \left(u_s^H\right)_k + \left(\partial_{u_s}^H T_{\boldsymbol{u}}^H\right)_k \left(\partial_{u_t}^H T_{\boldsymbol{u}}^H\right)_k \left(u_t^H\right)_k \\ &\quad - \left(\partial_{u_s}^H T_{\boldsymbol{u}}^H\right)_k \Big(\left(T_{\boldsymbol{u}}^H\right)_k - \left(R^H\right)_k\Big) \\ \left(\bar{G}_s^H\right)_k &= \left(\left(\partial_{v_s}^H R_{\boldsymbol{v}}^H\right)^2\right)_k \left(v_s^H\right)_k + \left(\partial_{v_s}^H R_{\boldsymbol{v}}^H\right)_k \left(\partial_{v_t}^H R_{\boldsymbol{v}}^H\right)_k \left(v_t^H\right)_k \\ &\quad - \left(\partial_{v_s}^H R_{\boldsymbol{v}}^H\right)_k \Big(\left(R_{\boldsymbol{v}}^H\right)_k - \left(T^H\right)_k\Big) \end{aligned} \tag{3.43}$$

for $s, t = 1, 2$, $s \neq t$ and $k = (j-2)(n-1) + (i-1)$ for $i, j = 2, \ldots, n-1$. We then solve the matrix equation (3.41) using a direct method, that is we solve

$$\boldsymbol{w}^H = \left(\bar{\boldsymbol{A}}^H\right)^{-1} \bar{\boldsymbol{F}}^H \tag{3.44}$$

Second Proposed Coarsest Level Solver C2: Similar to what we did for **C1**, we can express the system (2.36) on the coarsest grid in the following matrix form

$$\tilde{\boldsymbol{A}}^H \boldsymbol{w}^H = \bar{\boldsymbol{F}}^H \tag{3.45}$$

where $\tilde{\boldsymbol{A}}^H \in \mathbb{R}^{4(n-2)^2 \times 4(n-2)^2}$ has the following structure

$$\tilde{\boldsymbol{A}}^H = \begin{pmatrix} \boldsymbol{A}_1^H & \boldsymbol{0} & \boldsymbol{I}_2 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{A}_2^H & \boldsymbol{0} & \boldsymbol{I}_2 \\ \boldsymbol{I}_2 & \boldsymbol{0} & \boldsymbol{B}_1^H & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I}_2 & \boldsymbol{0} & \boldsymbol{B}_2^H \end{pmatrix} \tag{3.46}$$

where $\boldsymbol{A}_m^H, \boldsymbol{B}_m^H \in \mathbb{R}^{(n-2)^2 \times (n-2)^2}$ and $\boldsymbol{u}_m^H, \boldsymbol{v}_m^H, \bar{\boldsymbol{F}}_m^H, \bar{\boldsymbol{G}}_m^H \in \mathbb{R}^{(n-2)^2 \times 1}$ have the same structure as shown in **C1**, with RHS terms $\left(\bar{F}_m^H\right)_k^{(l+1)}, \left(\bar{G}_m^H\right)_k^{(l+1)}$ given by

$$\begin{aligned} \left(\bar{F}_m^H\right)_k &= \left(\left(\partial_{u_m}^H T_{\boldsymbol{u}}^H\right)^2\right)_k \left(u_m^H\right)_k - \left(\partial_{u_m}^H T_{\boldsymbol{u}}^H\right)_k \Big(\left(T_{\boldsymbol{u}}^H\right)_k - \left(R^H\right)_k\Big) \\ \left(\bar{G}_m^H\right)_k &= \left(\left(\partial_{v_m}^H R_{\boldsymbol{v}}^H\right)^2\right)_k \left(v_m^H\right)_k - \left(\partial_{v_m}^H R_{\boldsymbol{v}}^H\right)_k \Big(\left(R_{\boldsymbol{v}}^H\right)_k - \left(T^H\right)_k\Big). \end{aligned} \tag{3.47}$$

Again we solve the matrix equation (3.45) in a similar way to that shown in **C1**.

4 Numerical results

Now we will present some experimental results comparing three models, these are

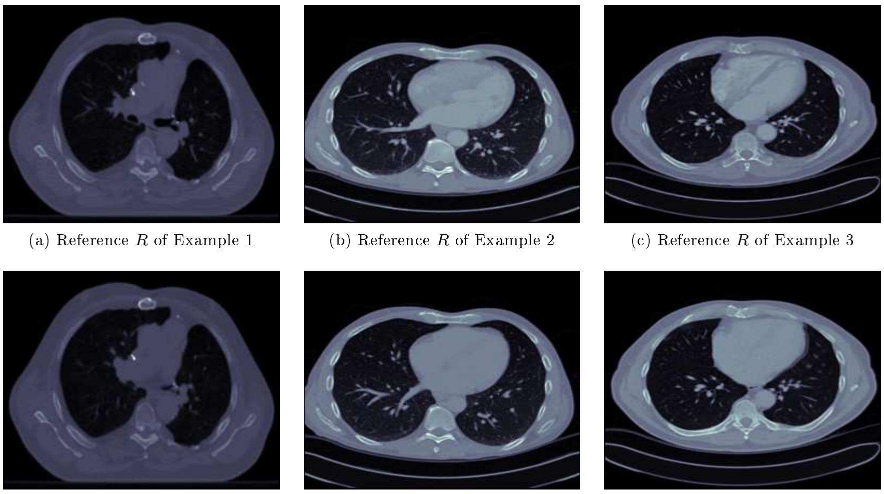
(i) A NMG scheme, similar to our proposed scheme, applied to a standard unidirectional diffusion model which we denote by **DNMG**.

(ii) Our proposed NMG applied to our inverse consistent model , equipped with smoother **S2** and solver **C1**, which we denote by **ICNMG1**.

(iii) Our proposed NMG applied to our inverse consistent model , equipped with smoother **S3** and solver **C2**, which we denote by **ICNMG2**.

Using these results we will demonstrate how our new **ICNMG** models produce comparable results, both visually and numerically, to the **DNMG** model while maintaining non-folding results even in the case of a 'bad' parameter choice. In addition we will also show how our simplified smoother **S3** in **ICNMG2** improves upon the CPU time, while maintaining the same level of accuracy, compared with **ICNMG1** which uses the fully coupled smoother **S2**.

In order to gain a qualitative measure in the accuracy between the two models, we choose to use the structural similarity (SSIM) [41] and relative errors $Err_F = \frac{\|T_u - R\|_2^2}{\|R\|_2^2}$, $Err_B = \frac{\|R_v - T\|_2^2}{\|T\|_2^2}$ corresponding to the forward and backward transformations respectively. Additionally in [11] it was shown that the quantity $Q_{min} = \det(\nabla \boldsymbol{\varphi})$ can be used to indicate the presence of folding if $Q_{min} \leq 0$, likewise if $Q_{min} > 0$ this indicates that no folding is present. Moreover, we will consider the NMG method to have converged only if one of the following criteria have been met; Average relative residual reaches $\varepsilon_1 = 10^{-2}$, maximum relative residual reaches $\varepsilon_2 = 10^{-2}$ or the number of NMG cycles reaches $\varepsilon_3 = 15$. It should also be noted that for our proposed **ICNMG** models, we only consider the NMG to have converged it both the forward and backward problems have converged according to the above stopping criteria. For all models we select the weighting parameter $\alpha = \frac{1}{15}$, and in our **ICNMG** models we set the second parameter to be $\beta = 10^4$. We performed our experiments on 3 sets of real lung CT images as shown in Figure 2. We also note that in Tables 2-8 green Q_{min} values indicate no folding in the transformation, while red values indicate folding is present in the transformation.



(a) Reference R of Example 1 (b) Reference R of Example 2 (c) Reference R of Example 3

(d) Template T of Example 1 (e) Template T of Example 2 (f) Template T of Example 3

Figure 2: Three Pairs of Test Images.

Example 1 Results: From Figure 3 we see that the **DNMG** model, as well as our **ICNMG** models, produce visually very similar results. This trend is backed up further by the results shown in Table 2, where we see near identical SSIM and relative error values. In addition we see that our **ICNMG** models produce larger CPU times when compared with the **DNMG** model, however this increase is to be expected since our **ICNMG** models must solve additional equations. Moreover we also see that our simplified smoother **S3**, which is used in our **ICNMG** model, produces noticeably smaller CPU times when compared with out **ICNMG1** model which uses the fully couple smoother **S2** while maintaining the same level of accuracy. Also since our **ICNMG** models require both forward and backward problems to converge, we see a slight increase in the number of NMG cycles required when compared with the **DNMG** model. This pattern of results is also seen in Table 3 where again all 3 models produce similar results with our **ICNMG** models requiring an additional NMG cycle to converge plus larger CPU times, with our **ICNMG2** model being significantly faster than our **ICNMG1** model. In all cases we see that all models produce positive Q_{min} values which indicates no folding is present in the transformations.

Example 2 Results: In Example 2, wee see the same pattern of results that we did for Example 1. Namely near identical results both visually (Figure 4) and numerically (Tables 4 and 5) with larger CPU times for our **ICNMG** models, and our **ICNMG2** model much faster than our **ICNMG1** model. In addition all 3 models produce non-folding results in all cases. However when considering the 'bad' parameter case $\alpha = \frac{1}{25}$ in Table 6, we see that the **DNMG** model produces negative Q_{min} values in 3 out of the 4 cases whereas both of our **ICNMG** models maintain the physical integrity of the transformation while achieving the same level of accuracy in all 4 cases. An example of how the mesh plots of the transformations from the **DNMG** model and our **ICNMG2** model for the 128^2 example from Table 6 can be seen in Figure 1. Here we see that the mesh from our **ICNMG2** model is much smoother than that from the **DNMG** model. We remark that the **DNMG** model can be modified to also produce non-folding by resetting the NMG scheme with a larger parameter α if folding occurs, however this solution extremely expensive computationally in addition to producing less accurate registration results in terms of SSIM and error values.

Example 3 Results: From Figure 5 and Tables 7 and 8 we see the same trend in results that was present in Examples 1 and 2, while we again see all cases produce non-folding transformations.

Testing of sensitivity of parameters for ICNMG2 model: Here we perform a test on how robust our **ICNMG2** model is to the choice of parameters α and β. To do this we tracked the $SSIM$ and Q_{min} values across a total of 25 different sets of parameter values, that is all combinations resulting from the parameters $\alpha = \frac{1}{10}, \frac{1}{15}, \frac{1}{20}, \frac{1}{25}, \frac{1}{30}$ and $\beta = 0, 10^3, 10^4, 10^5, 10^6$, and can be seen in Figures 6 and 7 respectively. In addition we remark that we have included a simulation for the **DNMG** model in our tests by considering the parameter $\beta = 0$. From Figure 6 we see that our **ICNMG2** model maintains very similar $SSIM$ values when compared with the **DNMG** model ($\beta = 0$ column), and there is little variation in the values as the parameter β is varied in our **ICNMG2** model. However the advantage of our **ICNMG** is shown more clearly in Figure 7 where we have tracked the Q_{min} values across the different parameter tests, here red indicates $Q_{min} < 0$ while green indicates $Q_{min} > 0$. From this figure we see that our **ICNMG2** is robust to folding for a much larger range of α values when compared with the diffusion model which has a much more limited range of viable α choices.

5 Conclusions

In this paper we first explained how many standard variational registration models do no place any emphasis on maintaining the physical accuracy of the transformations, thus potentially leading to physically inaccurate transformations with folding. Next we explained how inverse consistent models, such as the Christensen-Johnson model proposed in [15], can help improve robustness to folding. We also mentioned how the model in [15] is impractical for real medical image problems owing to the extensive computational cost resulting from solving the associated minimisation problem. In order to help avoid this problem, we first proposed a linearisation of the inverse consistency constraint from the Christensen-Johnson model to remove the additional non-linearities arising from this term when compared with typical diffusion type models, as well as alleviating the computational cost of directly computing the inverse displacements. Next we proposed the use of a fast NMG framework, based upon the scheme proposed by Chumchob-Chen in [19], along with 3 potential smoother schemes to further reduce the computational workload of the proposed inverse consistent model. In addition we also performed an analysis of the 3 proposed smoothers to determine their suitability for use in the NMG scheme, and how they can impact the convergence of the NMG. Next we showed, using 3 sets of real lung CT images, how our proposed inverse consistent model maintains the same level of accuracy as a unidirectional diffusion model using a similar NMG scheme, while being robust to parameter choice and folding even in the case of a 'bad' weighting parameter value which causes folding in the transformation obtained from the diffusion model.

Image Size n^2	**Initial**	**DNMG**	**ICNMG1**	**ICNMG2**
	$SSIM/Err_F(\%)$	$SSIM/Err_F\,(\%)/NMG/CPU\,(s)/Q_{min}$	$SSIM/Err_F\,(\%)/NMG/CPU\,(s)/Q_{min}$	$SSIM/Err_F\,(\%)/NMG/CPU\,(s)/Q_{min}$
128^2	0.915/0.35	0.938/0.22/1/0.167/0.553	0.938/0.22/2/1.498/0.489	0.938/0.22/2/0.879/0.489
256^2	0.914/0.38	0.935/0.27/1/0.822/0.673	0.932/0.28/2/5.155/0.654	0.933/0.28/2/3.031/0.654
512^2	0.939/0.37	0.953/0.27/1/4.082/0.669	0.949/0.28/2/24.557/0.658	0.949/0.28/2/14.180/0.658
1024^2	0.958/0.37	0.967/0.27/1/18.818/0.667	0.964/0.29/2/111.034/0.656	0.964/0.29/2/66.814/0.656

Table 2: Example 1: Comparison of forward registrations between 3 methods on different image sizes.

Image Size n^2	**Initial**	**DNMG**	**ICNMG1**	**ICNMG2**
	$SSIM/Err_B(\%)$	$SSIM/Err_B\,(\%)/NMG/CPU\,(s)/Q_{min}$	$SSIM/Err_B\,(\%)/NMG/CPU\,(s)/Q_{min}$	$SSIM/Err_B\,(\%)/NMG/CPU\,(s)/Q_{min}$
128^2	0.915/0.34	0.940/0.17/1/0.204/0.654	0.939/0.22/2/1.498/0.786	0.939/0.22/2/0.879/0.786
256^2	0.914/0.37	0.936/0.22/1/0.874/0.573	0.934/0.27/2/5.155/0.718	0.934/0.27/2//3.031/0.719
512^2	0.939/0.36	0.953/0.22/1/4.046/0.639	0.949/0.27/2/24.557/0.695	0.949/0.27/2/14.180/0.695
1024^2	0.958/0.36	0.968/0.22/1/17.935/0.633	0.965/0.28/2/111.034/0.686	0.965/0.28/2/66.814/0.686

Table 3: Example 1: Comparison of backward registrations between 3 methods on different image sizes.

Image Size n^2	**Initial**	**DNMG**	**ICNMG1**	**ICNMG2**
	$SSIM/Err_F(\%)$	$SSIM/Err_F\,(\%)/NMG/CPU\,(s)/Q_{min}$	$SSIM/Err_F\,(\%)/NMG/CPU\,(s)/Q_{min}$	$SSIM/Err_F\,(\%)/NMG/CPU\,(s)/Q_{min}$
128^2	0.808/1.02	0.892/0.37/2/0.415/0.451	0.891/0.37/2/1.582/0.353	0.890/0.37/2/0.640/0.241
256^2	0.767/1.07	0.871/0.40/2/1.512/0.250	0.868/0.42/2/5.202/0.157	0.868/0.42/2/3.025/0.024
512^2	0.779/1.08	0.868/0.41/2/6.819/0.519	0.866/0.43/2/24.572/0.423	0.866/0.43/2/14.232/0.423
1024^2	0.828/1.08	0.892/0.40/2/31.895/0.520	0.891/0.43/2//111.561/0.413	0.891/0.43/2/66.537/0.413

Table 4: Example 2: Comparison of forward registrations between 3 methods on different image sizes.

Image Size n^2	**Initial**	**DNMG**	**ICNMG1**	**ICNMG2**
	$SSIM/Err_B(\%)$	$SSIM/Err_B\,(\%)/NMG/CPU\,(s)/Q_{min}$	$SSIM/Err_B\,(\%)/NMG/CPU\,(s)/Q_{min}$	$SSIM/Err_B\,(\%)/NMG/CPU\,(s)/Q_{min}$
128^2	0.808/1.00	0.886/0.36/2/0.479/0.361	0.886/0.36/2/1.582/0.155	0.885/0.36/2/0.640/0.073
256^2	0.767/1.05	0.861/0.38/2/1.561/0.212	0.861/0.41/2/5.202/0.220	0.860/0.41/2/3.025/0.167
512^2	0.779/1.06	0.862/0.40/2/7.054/0.419	0.861/0.42/2/24.572/0.366	0.861/0.42/2/14.232/0.366
1024^2	0.828/1.06	0.889/0.40/2/31.370/0.405	0.890/0.42/2/111.561/0.350	0.890/0.42/2/66.537/0.350

Table 5: Example 2: Comparison of backward registrations between 3 methods on different image sizes.

Image Size n^2	**Initial**	**DNMG**	**ICNMG1**	**ICNMG2**
	$SSIM/Err_F(\%)$	$SSIM/Err_F\,(\%)/NMG/CPU\,(s)/Q_{min}$	$SSIM/Err_F\,(\%)/NMG/CPU\,(s)/Q_{min}$	$SSIM/Err_F\,(\%)/NMG/CPU\,(s)/Q_{min}$
128^2	0.808/1.02	0.872/0.36/2/0.426/−0.245	0.896/0.36/2/1.521/0.360	0.886/0.36/2/0.821/0.114
256^2	0.767/1.07	0.855/0.32/4/2.182/−0.374	0.874/0.36/2/5.255/0.220	0.871/0.36/2/3.355/0.316
512^2	0.779/1.08	0.876/0.34/2/6.907/−0.141	0.872/0.36/2/24.525/0.098	0.871/0.36/2/15.225/0.214
1024^2	0.828/1.08	0.900/0.32/2/33.889/0.214	0.896/0.36/2/111.118/0.168	0.895/0.36/2/73.118/0.240

Table 6: Example 2: Comparison of forward registrations between 3 methods on different image sizes for a 'bad' parameter value $\alpha = \frac{1}{25}$.

Image Size n^2	**Initial**	**DNMG**	**ICNMG1**	**ICNMG2**
	$SSIM/Err_F(\%)$	$SSIM/Err_F\,(\%)/NMG/CPU\,(s)/Q_{min}$	$SSIM/Err_F\,(\%)/NMG/CPU\,(s)/Q_{min}$	$SSIM/Err_F\,(\%)/NMG/CPU\,(s)/Q_{min}$
128^2	0.847/0.94	0.908/0.34/2/0.324/0.230	0.910/0.37/2/1.414/0.259	0.900/0.39/2/0.646/0.169
256^2	0.805/1.05	0.899/0.31/2/1.418/0.513	0.897/0.32/2/5.147/0.467	0.896/0.32/2/3.007/0.416
512^2	0.805/1.08	0.884/0.32/2/6.941/0.481	0.882/0.32/2/24.795/0.491	0.882/0.32/2/14.195/0.490
1024^2	0.842/1.08	0.901/0.32/2/33.210/0.411	0.902/0.32/2/111.887/0.589	0.902/0.32/2/66.789/0.588

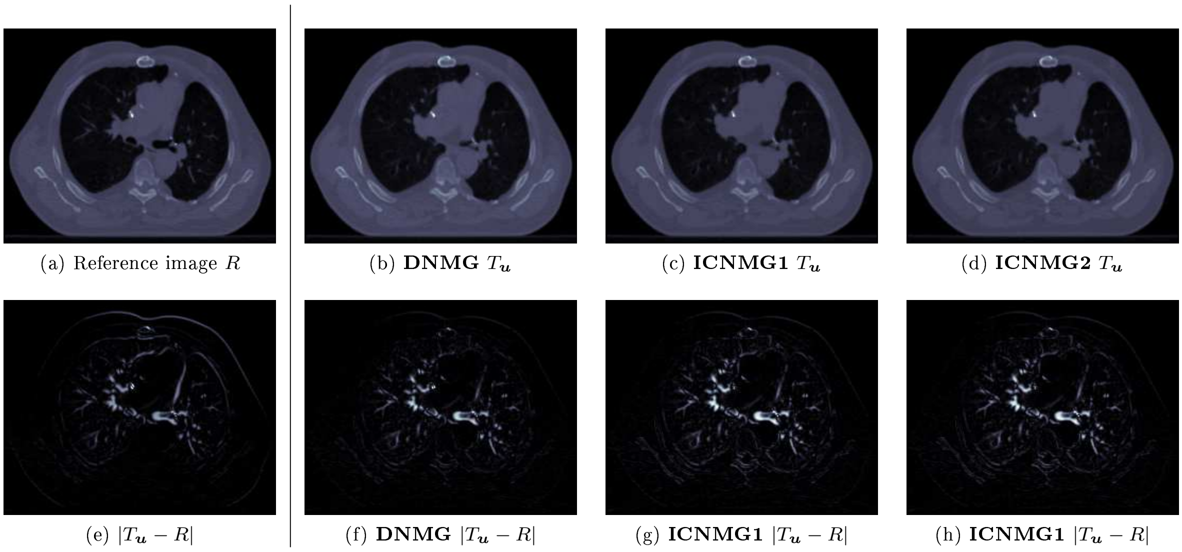
Table 7: Example 3: Comparison of forward registrations between 3 methods on different image sizes.

Image Size n^2	**Initial**	**DNMG**	**ICNMG1**	**ICNMG2**
	$SSIM/Err_B(\%)$	$SSIM/Err_B\,(\%)/NMG/CPU\,(s)/Q_{min}$	$SSIM/Err_B\,(\%)/NMG/CPU\,(s)/Q_{min}$	$SSIM/Err_B\,(\%)/NMG/CPU\,(s)/Q_{min}$
128^2	0.847/1.01	0.915/0.35/2/0.391/0.350	0.912/0.40/2/1.414/0.168	0.904/0.42/2/0.646/0.012
256^2	0.805/1.12	0.899/0.34/2/1.485/0.525	0.899/0.34/2/5.147/0.489	0.898/0.34/2/3.007/0.461
512^2	0.805/1.16	0.882/0.34/2/6.930/0.467	0.882/0.35/2/24.795/0.416	0.882/0.35/2/14.195/0.416
1024^2	0.842/1.16	0.899/0.34/2/33.301/0.440	0.902/0.35/2/111.887/0.435	0.902/0.35/2/66.789/0.435

Table 8: Example 3: Comparison of backward registrations between 3 methods on different image sizes.

Image Size n^2	Image Example	α	DNMG		ICNMG1		ICNMG2	
			CPU (s)	Ratio	CPU (s)	Ratio	CPU (s)	Ratio
128^2	Example 2 (Forward)	$\frac{1}{15}$	0.415	–	1.582	–	0.640	
256^2			1.512	3.643	5.202	3.288	3.025	4.727
512^2			6.819	4.510	24.572	4.724	14.232	4.705
1024^2			31.895	4.677	111.561	4.540	66.537	4.675

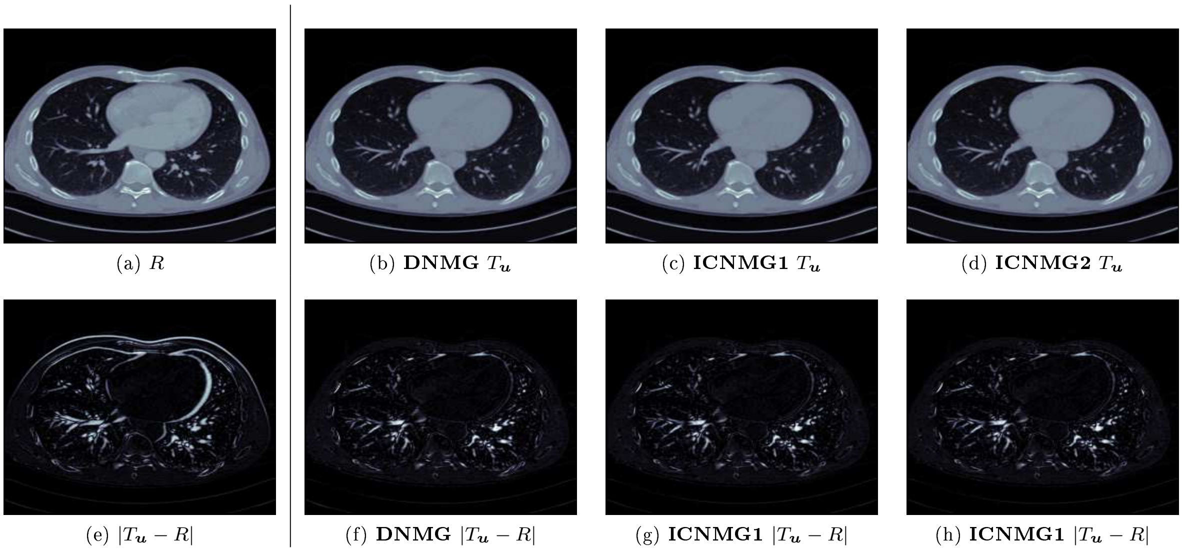
Table 9: Test on optimal complexity in CPU time ratio for 2 NMG methods. The optimal ratio is approximately 4.5 for an $O(NlogN)$ NMG method (with $N = n^2$).



(a) Reference image R (b) **DNMG** $T_{\boldsymbol{u}}$ (c) **ICNMG1** $T_{\boldsymbol{u}}$ (d) **ICNMG2** $T_{\boldsymbol{u}}$

(e) $|T_{\boldsymbol{u}} - R|$ (f) **DNMG** $|T_{\boldsymbol{u}} - R|$ (g) **ICNMG1** $|T_{\boldsymbol{u}} - R|$ (h) **ICNMG1** $|T_{\boldsymbol{u}} - R|$

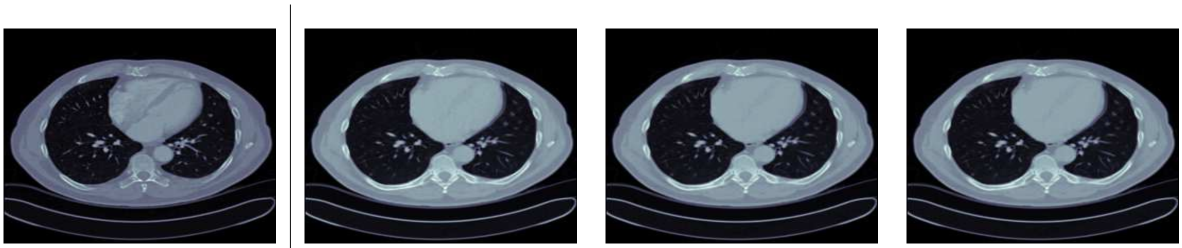
Figure 3: Example 1: Registration of $2(a)$ and $2(d)$ of size 256×256 by 3 methods with initial error shown by image (e). Images (b), (c) and (d) show the deformed template images obtained using the **DNMG**, **ICNMG1** and **ICNMG2** models respectively, while images (f), (g) and (h) show the respective final errors.



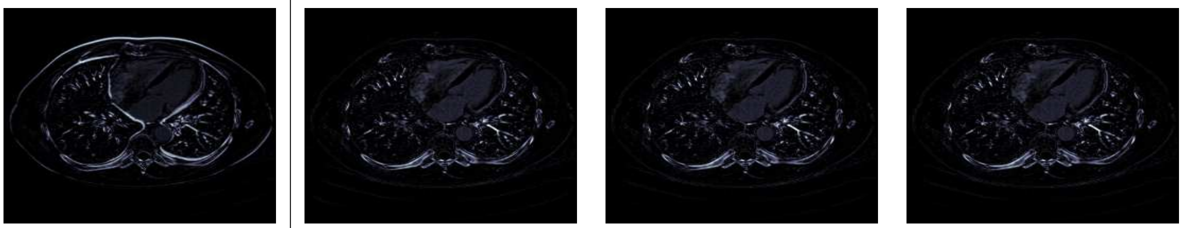
(a) R (b) **DNMG** $T_{\boldsymbol{u}}$ (c) **ICNMG1** $T_{\boldsymbol{u}}$ (d) **ICNMG2** $T_{\boldsymbol{u}}$

(e) $|T_{\boldsymbol{u}} - R|$ (f) **DNMG** $|T_{\boldsymbol{u}} - R|$ (g) **ICNMG1** $|T_{\boldsymbol{u}} - R|$ (h) **ICNMG1** $|T_{\boldsymbol{u}} - R|$

Figure 4: Example 2: Registration of $2(b)$ and $2(e)$ of size 256×256 by 3 methods with initial error shown by image (e). Images (b), (c) and (d) show the deformed template images obtained using the **DNMG**, **ICNMG1** and **ICNMG2** models respectively, while images (f), (g) and (h) show the respective final errors.

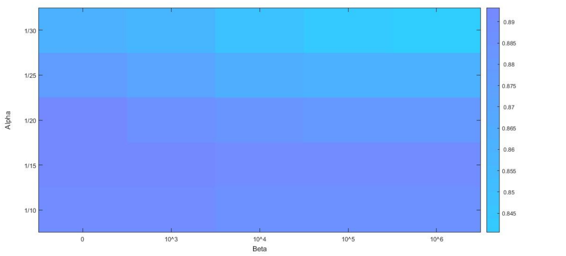


(a) R (b) **DNMG** $T_{\boldsymbol{u}}$ (c) **ICNMG1** $T_{\boldsymbol{u}}$ (d) **ICNMG2** $T_{\boldsymbol{u}}$

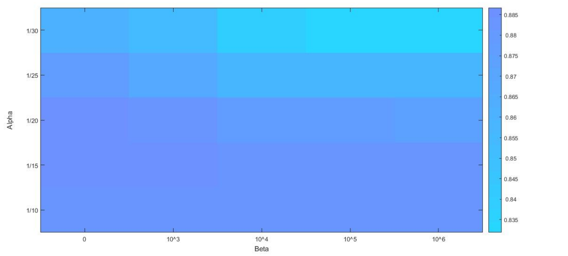


(e) $|T_{\boldsymbol{u}} - R|$ (f) **DNMG** $|T_{\boldsymbol{u}} - R|$ (g) **ICNMG1** $|T_{\boldsymbol{u}} - R|$ (h) **ICNMG2** $|T_{\boldsymbol{u}} - R|$

Figure 5: Example 3: Registration of $2(c)$ and $2(f)$ of size 256×256 by 3 methods with initial error shown by image (e). Images (b), (c) and (d) show the deformed template images obtained using the **DNMG**, **ICNMG1** and **ICNMG2** models respectively, while images (f), (g) and (h) show the respective final errors.

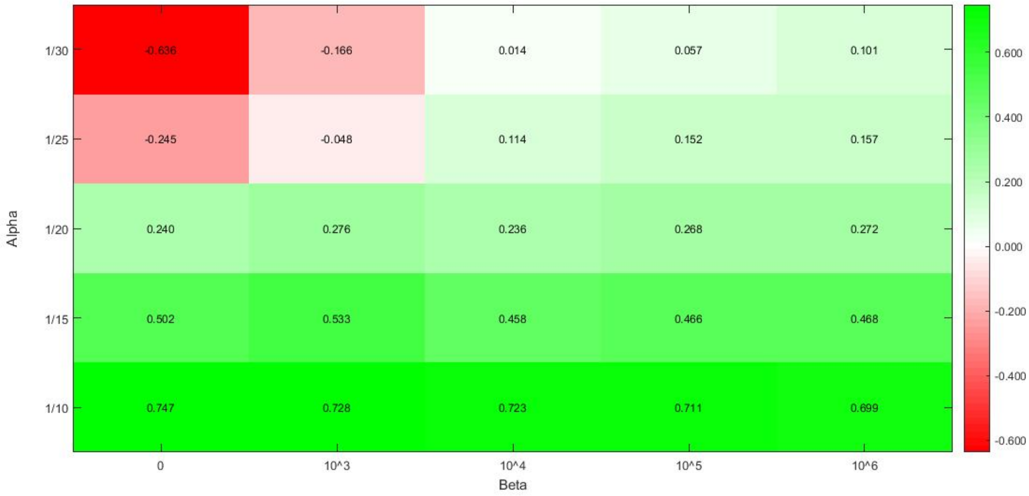


(a) Heat map of $SSIM$ values over a range of parameter choices α, β for the forward problem

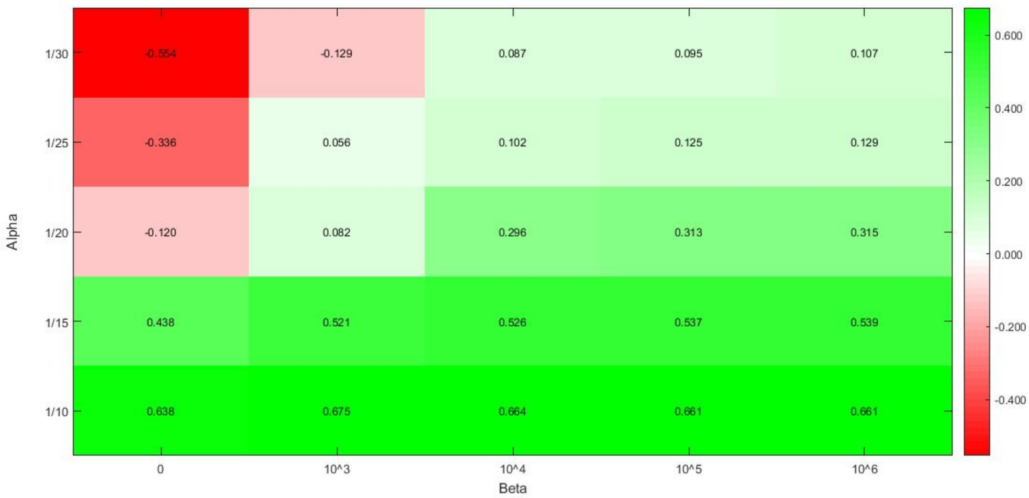


(b) Heat map of $SSIM$ values over a range of parameter choices α, β for the backward problem

Figure 6: Comparison of how the $SSIM$ values vary with different choices of the parameters α and β for Example 2.



(a) Heat map of Q_{min} values over a range of parameter choices α, β for the forward problem



(b) Heat map of Q_{min} values over a range of parameter choices α, β for the backward problem

Figure 7: Comparison of how the Q_{min} values vary with different choices of the parameters α and β for Example 2.

References

[1] G. Auzias, O. Colliot, J.A Glaunès, M. Perrot, J.F. Mangin, A. Trouvé, and S. Baillet. Diffeomorphic brain registration under exhaustive sulcal constraints. *IEEE Transactions on Medical Imaging*, 30(6):1214–1227, 2011.

[2] R. Bajscy and S. Kovačič. Multiresolution elastic matching. *Comp. Vision Graph.*, 46(1):1–21, 1989.

[3] M. Bazargani, A. Anjos, F. G. Lobo, A. Mollahosseini, and H. R. Shahbazkia. Affine image registration transformation estimation using a real coded genetic algorithm with SBX. *CoRR*, abs/1204.2139, 2012.

[4] A. Brandt. Multilevel adaptive solutions to BVPs. *Math. Comp.*, 31:333–390, 1977.

[5] C. Broit. *Optimal registration of deformed images.* PhD thesis, University of Pennsylvania, 1981.

[6] T. Brox, C. Bregler, and J. Malik. Large displacement optical flow. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2009.

[7] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. *ECCV*, 3024:25–36, 2004.

[8] T. Brox and J. Malik. Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33:500–513, 2011.

[9] A. Bruhn, J. Weickert, C. Feddern, T. Kohlberger, and C. Schnörr. Real-time optic flow computation with variational methods. *Computer Analysis of Images and Patterns*, 2756:222–229, 2003.

[10] A. Bruhn, J. Weickert, C. Feddern, T. Kohlberger, and C. Schnörr. Variational optic flow computation in real-time. *IEEE Transactions on Image Processing*, 14:608–615, 2006.

[11] M. Burger, J. Modersitzki, and L. Ruthotto. A hyperelastic regularization energy for image registration. *SIAM Journal on Scientific Computing*, 35(1):B132–B148, 2013.

[12] K. Cao, G.E. Christensen, K. Ding, K. Du, M.L. Raghavan, R.E. Amelon, K.M. Baker, E.A. Hoffman, and J.M. Reinhardt. Tracking regional tissue volume and function change in lung using image registration. *Int. Journal of Biomedical Imaging*, Article ID 956248 (OA), 2012.

[13] K. Chen. *Matrix Preconditioning Techniques and Applications.* Cambridge University Press, 2005.

[14] Y. M. Chen and X. J. Ye. *The Legacy of Alladi Ramakrishnan in the Mathematical Sciences*, chapter on: Inverse consistent deformable image registration, pages 419–440. Springer, New York, 2010.

[15] G.E. Christensen and H.J. Johnson. Consistent image registration. *IEEE Transactions on Medical Imaging*, 20(7):568–582, 2001.

[16] G.E. Christensen, S.C. Joshi, and M.I. Miller. Volumetric transformation of brain anatomy. *IEEE Transactions on Medical Imaging*, 16(6):864–877, 1997.

[17] G.E. Christensen, J.H. Song, W. Lu, I. El Naqa, and D.A. Low. Tracking lung tissue motion and expansion/compression with inverse consistent image registration and spirometry. *Medical Physics*, 34(6):2155–2163, 2007.

[18] N. Chumchob and K. Chen. A robust affine image registration method. *International Journal of Numerical Analysis and Modelling*, 6(2):311–334, 2009.

[19] N. Chumchob and K. Chen. A robust multigrid approach for variational image registration models. *Journal of Computational and Applied Mathematics*, 236(5):653–674, 2011.

[20] N. Chumchob, K. Chen, and C. Brito-Loeza. A fourth order variational image registration model and its fast multigrid algorithm. *Multiscale Moddeling and Simulation*, 9(1):89–128, 2010.

[21] B. Dacorogna. *Direct methods in the calculus of variations.* Springer-Verlag, 1989.

[22] O. Demetz, M. Stoll, S. Volz, J. Weickert, and A. Bruhn. Learning brightness transfer functions for the joint recovery of illumination changes and optical flow. In *ECCV 2014*, pages 455–471, 2014.

[23] C. Frohn-Schauf, S. Henn, L.Hömke, and K. Witsch. Total variation based image registration. In *International Conference on PDE-Based Image Processing and Related Inverse Problems Series: Mathematics and Visualization*, pages 305–323. Springer Verlag, 2006.

[24] C. Frohn-Schauf, S. Henn, and K. Witsch. Multigrid based total variation image registration. *Computing and Visualization in Science*, 11(2):101–113, 2008.

[25] A. Gooya, G. Biros, and C. Davatzikos. Deformable registration of glioma images using em algorithm and diffusion reaction modelling. *IEEE Transactions on Medical Imaging*, 30(2):375–390, 2011.

[26] V. Gorbunova, J. Sporring, P. Lo, M. Loeve, H.A. Tiddens, M. Nielsen, A. Dirksen, and M. de Bruijne. Mass preserving image registration for lung CT. *Medical Image Analysis*, 16(4):786–795, 2012.

[27] N.M. Grosland, R. Bafna, and V.A. Magnotta. Automated hexahedral meshing of anatomic structures using deformable registration. *Computer Methods in Biomechanics and Biomedical Engineering*, 12(1):35–43, 2009.

[28] T. Guerro, K. Sanders, E. Castillo, Y. Zhang, L. Bidaut, T. Pan, and R. Komaki. Dynamic ventillation imaging from four-dimensional computed tomography. *Phys Med Biol.*, 51(4):777–791, 2006.

[29] Christoph Guetter, Hui Xue, Christophe Chefd'hotel, and Jens Guehring. Efficient symmetric and inverse-consistent deformable registration through inter-leaved optimization. *IEEE International Symposium on Biomedical Imaging From Nano to Macro*, 2011.

[30] E. Haber and J. Modersitzki. A multilevel method for image registration. *SIAM J. Sci. Comput.*, 27(5):1594–1607, 2006.

[31] S. Henn. A multigrid method for a fourth-order diffusion equation with application to image processing. *SIAM Journal on Scientific Computing*, 27(3):831–849, 2005.

[32] S. Henn and K. Witsch. Iterative multigrid regularization techniques for image matching. *SIAM Journal on Scientific Computing*, 23(4):1077–1093, 2001.

[33] D.L.G. Hill, P.G. Batchelor, M. Holden, and D.J. Hawkes. Medical image registration. *Physics in medicine and biology*, 46(3):1–45, 2001.

[34] H. J. Johnson and G. E. Christensen. Consistent landmark and intensity-based image registration. *IEEE Transactions on Medical Imaging*, 21(5):450–461, 2002.

[35] H. Köstler, K. Ruhnau, and R. Wienands. Multigrid solution of the optical flow system using a combined diffusion- and curvature-based regularizer. *Numerical Linear Algebra with Applications*, 15(2-3):201–218, 2008.

[36] K. C. Lam and L. M. Lui. Landmark-and intensity-based registration with large deformations via quasi-conformal maps. *SIAM Journal on Imaging Sciences*, 7(4):2364–2392, 2014.

[37] T. Lin, C. Le Guyader, I.D. Dinov, P.M. Thompson, A.W. Toga, and L.A. Vese. Gene expression data to mouse atlas registration using a nonlinear elasticity smoother and landmark points constraints. *J. Sci. Comput.*, 50:586–609, 2012.

[38] J. Modersitzki. *Numerical Methods for Image Registration*. Oxford University Press, 2004.

[39] J. Modersitzki. *Flexible Algorithms for Image Registration*. SIAM publications, 2009.

[40] A. Pevsner, B. Davis, S. Joshi, A. Hertanto, J. Mechalakos, E. Yorke, K. Rosenzweig, S. Nehmeh, Y.E. Erdi, J.L. Humm, S. Larson, C.C. Ling, and G.S. Mageras. Evaluation of an automated deformable image matching method for quantifying lung motion in respiration-correlated ct images. *Medical Physics*, 33(2):369–376, 2006.

[41] 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.

[42] M. Reuter, H. Rosas, and B.Bacth Fischl. Highly accurate inverse consistent registration: a robust approach. *Neuroimage*, 53 (4):1181–1196, 2010.

[43] L. Ruthotto, C. Greif, and J. Modersitzki. A stabilized multigrid solver for hyperelastic image registration. *Numerical Linear Algebra with Applications*, 24(5), 2017.

[44] D. Sarrut, V. Boldea, S. Miguet, and C. Ginestet. Simulation of four-dimensional ct images from deformable registration between inhale and exhale breath-hold ct scans. *Medical Physics*, 33(3):605–617, 2006.

[45] U. Trottenberg, C. Oosterlee, and A. Schüller. *Multigrid*. Academic Press, 2001.

[46] J. Wienands and W. Joppich. *Practical fourier analysis for multigrid method*. Chapman and Hall/CRC, USA, 2005.

[47] Deshan Yang, Hua Li, Daniel A Low, Joseph O Deasy, and Issam El Naqa. A fast inverse consistent deformable image registration method based on symmetric optical flow computation. *Physics in Medicine & Biology*, 53(21):6143, 2008.

[48] D. P. Zhang and K. Chen. A novel diffeomorphic model for image registration and its algorithm. *Journal Of Mathematical Imaging And Vision*, DOI: 10.1007/s10851-018-0811-3, 2018.