A robust multigrid approach for variational image registration models

Noppadol Chumchob\textsuperscript{a,b}, Ke Chen\textsuperscript{a,*}

\textsuperscript{a} Centre for Mathematical Imaging Techniques, Department of Mathematical Sciences, The University of Liverpool, Peach Street, Liverpool L69 7ZL, United Kingdom
\textsuperscript{b} Department of Mathematics, Faculty of Science, Silpakorn University, Nakorn Pathom 73000, Thailand

\textbf{ARTICLE INFO}

\textbf{MSC:}
65F10
65M55
68U10

\textbf{Keywords:}
Variational models
Deformable registration
Diffusion and curvature image models
Smoothing analysis
Nonlinear multigrid
Inverse problems

\textbf{ABSTRACT}

Variational registration models are non-rigid and deformable imaging techniques for accurate registration of two images. As with other models for inverse problems using the Tikhonov regularization, they must have a suitably chosen regularization term as well as a data fitting term. One distinct feature of registration models is that their fitting term is always highly nonlinear and this nonlinearity restricts the class of numerical methods that are applicable. This paper first reviews the current state-of-the-art numerical methods for such models and observes that the nonlinear fitting term is mostly 'avoided' in developing fast multigrid methods. It then proposes a unified approach for designing fixed point type smoothers for multigrid methods. The diffusion registration model (second-order equations) and a curvature model (fourth-order equations) are used to illustrate our robust methodology. Analysis of the proposed smoothers and comparisons to other methods are given. As expected of a multigrid method, being many orders of magnitude faster than the unilevel gradient descent approach, the proposed numerical approach delivers fast and accurate results for a range of synthetic and real test images.

\textcopyright 2011 Elsevier B.V. All rights reserved.

1. Introduction

Image registration is one of the most useful and yet challenging tasks in image processing applications. It is the process of finding an \textit{optimal geometric transformation} between \textit{corresponding} images. It can also be seen as the process of overlaying two or more images of the same or similar scene taken at different times, from different perspectives, and/or by different imaging machineries. Therefore, this procedure is required whenever a series of corresponding images needs to be compared or integrated. Applications that require a registration step range from art, astronomy, biology, chemistry, criminology, physics and remote sensing. Particularly, in medical applications, non-invasive imaging is increasingly used in almost all stages of patient care: from disease detection to treatment guidance and monitoring. For an overview on registration methodology, we refer to [1–3], and the references therein. This paper focuses on a robust multigrid approach for deformable image registration models in a variational formulation.

Variational partial differential equations (PDEs) based image registration models have been successfully proven to be very valuable tools in several applications, although much improvement is still required. A general framework of the image registration can be formulated as follows: given two images of the same object, respectively referred to as \textit{reference} \( R \) and \textit{template} \( T \), we search for a vector-valued transformation \( \varphi \) defined by

\[
\varphi(u)(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}^d, \quad \varphi(u)(x) : x \mapsto x + u(x)
\]
that depends on the unknown deformation or displacement field

\[ \mathbf{u} : \mathbb{R}^d \rightarrow \mathbb{R}^d, \quad \mathbf{u} : \mathbf{x} \mapsto \mathbf{u}(\mathbf{x}) = (u_1(\mathbf{x}), u_2(\mathbf{x}), \ldots, u_d(\mathbf{x}))^\top \]

such that the transformed template \( T \circ \varphi(\mathbf{u}(\mathbf{x})) = T(\mathbf{x} + \mathbf{u}(\mathbf{x})) \) becomes similar to the reference \( R \). Once the corresponding location \( \varphi(\mathbf{u}(\mathbf{x})) = \mathbf{x} + \mathbf{u}(\mathbf{x}) \) is calculated for each spatial location \( \mathbf{x} \) in the image domain \( \Omega \subset \mathbb{R}^d \), an image interpolation is required to assign the image intensity values for the transformed template \( T(\mathbf{u}) \) at non-grid locations within image boundaries. For locations outside the image boundaries, the image intensities are usually set to be a constant value, typically zero [3]. We see that the displacement \( \mathbf{u} \) measures by how much a point in the transformed template \( T(\mathbf{u}) \) has moved away from its original position in \( T \). Here we shall restrict ourselves to scalar or gray intensity images and model them as compactly supported functions mapping from the image domain \( \Omega \subset \mathbb{R}^d \) into \( V \subset \mathbb{R}^n \), where \( d \in \mathbb{N} \) represents the spatial dimension of the images which is usually \( d = 2 \) (planar images) or \( d = 3 \) (volume data set) with boundary \( \partial \Omega \). Without loss of generality we assume that the registration problem is described in the two-dimensional case \( (d = 2) \) throughout this work, but it is readily extendable to the three-dimensional case \( (d = 3) \). We also assume further that \( \Omega = [0, 1]^2 \subset \mathbb{R}^2 \) and \( V = [0, 1] \) for 2D gray intensity images.

Assume the image intensities of \( R \) and \( T \) are comparable (i.e. in a monomodal registration scenario), the task is to solve the minimization problem of a similarity measure

\[
\min_{\mathbf{u}} \left\{ D(\mathbf{u}) = \frac{1}{2} \int_\Omega (T(\mathbf{x} + \mathbf{u}(\mathbf{x})) - R(\mathbf{x}))^2 \, d\mathbf{x} \right\}, \tag{1}
\]

whose functional is the sum of squared differences (SSD) [4–20,3,21,22] which is widely used. As is known, this problem is generally ill posed in the sense of Hadamard. Therefore, the minimization of \( D \) will not guarantee an unique solution. It becomes necessary to impose a constraint on the solution \( \mathbf{u} \) via a deformation regularizer \( \mathcal{R} \) for penalizing unwanted and irregular solutions using some priori knowledge. In the Tikhonov regularization framework, the image registration problem can be posed as a minimization problem of the joint energy functional given by

\[
\min_{\mathbf{u}} \left\{ f_\alpha[\mathbf{u}] = D(\mathbf{u}) + \alpha \mathcal{R}(\mathbf{u}) \right\}, \tag{2}
\]

where \( \alpha > 0 \) is the regularization parameter that compromises similarity and regularity, and \( T(\mathbf{u}) = T(\mathbf{x} + \mathbf{u}(\mathbf{x})) \) represents the transformed template image. The regularizer \( \mathcal{R} \) is designed to ensure that the deformation field \( \mathbf{u} \) constructed is unique and close to the true solution. As the aim of this paper is to address the fast solutions issue, we consider two regularizers: firstly the diffusion regularizer as introduced in [4]:

\[
\mathcal{R}(\mathbf{u}) = \frac{1}{2} \sum_{l=1}^2 \int_\Omega |\nabla u_l|^2 \, d\mathbf{x}, \tag{3}
\]

and secondly the curvature regularizer in [5] as

\[
\mathcal{R}^\text{curv}(\mathbf{u}) = \frac{1}{2} \sum_{l=1}^2 \int_\Omega (\kappa_l(u_l))^2 \, d\mathbf{x} = \frac{1}{2} \sum_{l=1}^2 \int_\Omega (\Delta u_l)^2 \, d\mathbf{x}, \tag{4}
\]

as respective examples of second-order PDEs and fourth-order PDEs; our method will also be applicable to other models that lead to second-order PDEs e.g. the elastic and fluid models [3] and the total variation model [8,9], the combined regularization model [23] and other curvature models [24].

Although the multigrid techniques have been successfully used for numerical solutions for deformable image registration [9,10,12,14,16–18,21,22], none of the existing variants are optimal implementations. In these works, the nonlinear fitting term is not directly dealt with; we remark that other image restoration models [25–28] do not usually have nonlinearities resulting from a fitting term (unless a \( L_1 \) fitting or an expectation–maximization fitting is used). We may classify these existing variants into 2 categories:

1. **Linear multigrid framework.** Haber and Modersitzki [12] used an inexact Gauss–Newton (GN) scheme combined with a linear multigrid method as an inner solver for solving the elastic image registration problem corresponding to [2]. Henn [16] considered the curvature image registration problem and similarly used a coupled outer–inner iteration method with the inner solver provided by a linear multigrid method for solving the system of the fourth-order PDEs. Hörmke [17] concentrated on the elastic image registration and introduced a numerical solution of the minimization problem given by (2) using a regularized GN method with a trust region approach in which one normal equation corresponding to a linear subproblem is solved iteratively with a linear multigrid method. Köstler et al. [18] introduced a combined diffusion– and curvature-based regularizer for optical flow and deformable image registration problems and solved the resulting minimization problem represented in terms of (2) with an inexact GN method where each GN perturbation is estimated by a linear multigrid approach. Stürmer et al. [21] considered the diffusion image registration and solved the system of nonlinear PDEs using a coupled outer–inner iteration method with the inner solver provided by a linear multigrid method commonly used for heat equations.
2. The diffusion registration model and its numerical methods

We now introduce our first model and review briefly various solution methods paying particular attention to robustness of multigrid methods. The model itself is not particularly more important than other models from [3] but we use it to illustrate the fast solution issues of image registration. Below use the notation \( \partial_{x_i} F = \frac{\partial F}{\partial x_i} \) and \( \partial_{x_1 x_2} F = \frac{\partial^2 F}{\partial x_1 \partial x_2} \).

### 2.1. The diffusion model

The minimizer of the energy functional \( J_u \) in (2), defined by (1) and (3), with respect to the minimizer \( u = (u_1, u_2)^T \) satisfies the so-called Euler–Lagrange equations [3], given by the following system of two coupled, nonlinear, and elliptic partial differential equations (PDEs) for \( u = (u_1(x_1, x_2), u_2(x_1, x_2))^T \):

\[
\begin{align*}
N_1(u) &= -\alpha \Delta u_1 + \frac{1}{\partial_1 T(u)} \partial_{x_1} T(u) = 0, \\
N_2(u) &= -\alpha \Delta u_2 + \frac{1}{\partial_2 T(u)} \partial_{x_2} T(u) = 0,
\end{align*}
\]

subject to the homogeneous Neumann’s boundary conditions

\[
\partial_n u_1 = \partial_n u_2 = 0 \quad \text{on} \quad \partial \Omega.
\]

Here the nonlinear functions \( f_1(u), f_2(u) \) are from the data fitting terms, which are (as remarked) features of registration models distinct from other imaging models [26]. Refer to [9,4,10,14,3,21,22]. In fact, the nonlinear coupling of the two PDEs is through the term \( T(u) \). Hereby, \( \Delta \) denotes the Laplace operator, and \( n = (n_1, n_2)^T \) is the outward unit normal to the image boundary \( \partial \Omega \). Note that the first and second terms in (5) are the first variations of the regularizer term \( \mathcal{R} \) and the data term \( \mathcal{D} \), respectively.

### 2.2. Discretization by a finite difference method

For sake of clarity, let \( (u^h)_{i,j} = u^h(x_1, x_2) \) denote the grid function for \( l = 1, 2 \) with grid spacing \( h = (h_1, h_2) = (1/n_1, 1/n_2) \). Applying finite difference schemes based on the cell-centered grid points to discretize (5), the discrete Euler–Lagrange equations at a grid point \( (i, j) \) over the discrete domain,

\[
\Omega_h = \{ x \in \Omega | x = (x_1, x_2)^T = ((2i - 1)h_1/2, (2j - 1)h_2/2)^T, \ 1 \leq i \leq n_1, \ 1 \leq j \leq n_2 \}
\]

are given by

\[
\begin{align*}
N_1^h(u^h)_{i,j} &= -\alpha \Delta_h(u^h)_{i,j} + f_1^h(u^h)_{i,j} = g_{1,i,j}^h, \\
N_2^h(u^h)_{i,j} &= -\alpha \Delta_h(u^h)_{i,j} + f_2^h(u^h)_{i,j} = g_{2,i,j}^h
\end{align*}
\]

with the following notation
\[-L^h(u^h_{i,j}) = \frac{1}{h^2}((\Sigma)_{i,j}(u^h_{i,j}) - (\Sigma)_{i,j}(u^h_{i,j})) \]

\((\Sigma)_{i,j} = 2(1 + \gamma^2), \quad \gamma = h_1/h_2,\)

\((\Sigma)_{i,j}(u^h_{i,j}) = (u^h_{i,j})_{i+1,j} + (u^h_{i,j})_{i-1,j} + \gamma^2(u^h_{i,j+1}) + \gamma^2(u^h_{i,j-1})\)

\[f^h_1(u^h_1, u^h_2)_{i,j} = (T^h_{i,j} - R^h_{i,j})((T^h_{i+1,j} - T^h_{i-1,j})/(2h_1)),\]

\[f^h_2(u^h_1, u^h_2)_{i,j} = (T^h_{i,j} - R^h_{i,j})((T^h_{i,j+1} - T^h_{i,j-1})/(2h_2)),\]

\[T^h_{i,j} = \tau^h(i + (u^h_1)_{i,j}, j + (u^h_2)_{i,j}).\]

Here \(g^h_{i,j} = g^h_{2i,j} = 0\) on the finest grid in multigrid setting to be used shortly. We note that the approximations in (8) need to be adjusted at the image boundary \(\partial \Omega_h\) using the homogeneous Neumann boundary conditions

\[(u^h_1)_{i,1} = (u^h_1)_{i,2}, \quad (u^h_2)_{i,n_2} = (u^h_2)_{i,n_2-1}, \quad (u^h_1)_{1,j} = (u^h_1)_{2,j}, \quad (u^h_2)_{n_1,j} = (u^h_2)_{n_1-1,j}.\]  

(9)

2.3. Review of non-multigrid numerical solvers

The **first** commonly used method is a gradient descent approach solving, instead of the nonlinear elliptic system (5), the nonlinear parabolic system

\[
\begin{align*}
\partial_t u_1 - \alpha \Delta u_1 &= -(T(u) - R)\partial_t u_1 T(u), \\
\partial_t u_2 - \alpha \Delta u_2 &= -(T(u) - R)\partial_t u_2 T(u),
\end{align*}
\]

(10)

where \(u = (u(x,t), u_j(x,t))\) will converge to the solution of (5) when \(t \to \infty\), with the initial solution \(u(x,0)\), typically \(u(x,0) = 0\). The advantage is that various time-marching schemes can be used to solve (10) in order to circumvent the nonlinearity on the right-hand side. For example, the semi-implicit scheme can be proceeded as follows (in matrix vector form obtained from the discretized version of Eq. (10))

\[
\begin{align*}
[u^{(k+1)}_1](i) &= [I - \alpha \tau \sum_{l=1}^{2} A_l]^{-1} (u^{(k)}_1 - \tau f_1(u^{(k)}_1, u^{(k)}_2)) \\
[u^{(k+1)}_2](i) &= [I - \alpha \tau \sum_{l=1}^{2} A_l]^{-1} (u^{(k)}_2 - \tau f_2(u^{(k)}_1, u^{(k)}_2)).
\end{align*}
\]

(11)

Here, \(I\) is the identity matrix, \(f_1(u^{(k)}_1, u^{(k)}_2)\) is the discretized version of the second term in (5) using an appropriate finite difference approximation, usually the central finite difference, \(\tau > 0\) is the time-step determined by a forward difference approximation of the time derivative \(\partial_t u\), and \(A_l\) is the coefficient matrix from discretization of the Laplace operator \(\Delta\) along the \(l\)-coordinate direction subject to Neumann boundary conditions. The DCT-based method from [3] or the FT-based method from [33,34] may be used to solve the above equation.

The **second** method by an additive operator splitting (AOS) scheme introduced in [4,35,36] is faster and more efficient than the standard semi-implicit scheme (11). The basic idea is to replace the inverse of the sum by a sum of inverses. The corresponding iterations are then defined by

\[
\begin{align*}
[u^{(k+1)}_1](i) &= \frac{1}{2} \sum_{l=1}^{2} [(I - 2\alpha \tau A_l)^{-1} (u^{(k)}_1 - \tau f_1(u^{(k)}_1, u^{(k)}_2)) \\
[u^{(k+1)}_2](i) &= \frac{1}{2} \sum_{l=1}^{2} [(I - 2\alpha \tau A_l)^{-1} (u^{(k)}_2 - \tau f_2(u^{(k)}_1, u^{(k)}_2)).
\end{align*}
\]

(12)

which is much cheaper than those obtained from (11) because the two tri-diagonal systems in each component are solved per iteration rather than the 5-band system.

The **third** method by the one-cycle multi-resolution (or a coarse-to-fine FMG technique) is proposed in [20]. The idea is to solve (5) first with the Newton–Gauss–Seidel relaxation method given by

\[
\begin{align*}
u^{(new)}_1 &= u^{(old)}_1 - \frac{N_1(u^{(old)})}{\partial u_1 N_1(u^{(old)})} \\
u^{(new)}_2 &= u^{(old)}_2 - \frac{N_2(u^{(old)})}{\partial u_2 N_2(u^{(old)})}.
\end{align*}
\]

(13)
on the coarsest (lowest) resolution, and then interpolate the coarse solution with an appropriate method as a good initial guess solution to the next finer resolution. This process is repeated until (5) has been solved by (13) on the finest (highest) or original resolution. Although, this method provides well matched images in most of the cases and the basic idea is essentially the same as a FMG method as in [37,38,39,40], one should note that convergence cannot be proved even for much simpler equations [30]. This is due to the fact that the solution on the fine level depends strongly on the coarse one. As a consequence, errors introducing from the interpolation procedure may propagate in such a way that they spoil the overall results completely.

Although the above three methods are easy to implement on a unilevel and multilevel, they are not as efficient as a multigrid method.

2.4. Review of multigrid solvers and previous work

Multigrid techniques are widely used fast methods for various PDEs [37,38,41,30,39,40]. For deformable image registration models, we give a brief review here [9,10,12,14,16–18,21,22].

The basic idea of a multigrid method is to smooth high-frequency components of the error of the solution by performing a few steps with a smoother such that a smooth error term can be well represented and approximated on a coarser grid. After a linear or nonlinear residual equation has been solved on the coarse grid, a coarse-grid correction is interpolated back to the fine grid and used to correct the fine-grid approximation. Finally, the smoother is performed again in order to remove some new high-frequency components of the error introduced by the interpolation.

Below we shall first review the basic FAS-NMG algorithm before discussing previous work on solving (5). As it turns out, two approaches considered are both more efficient than an unilevel method but none are robust solvers.

2.4.1. The full approximation scheme (FAS)

For a nonlinear problem, the use of a full approximation scheme (the nonlinear multigrid method (NMG) in [29]) is natural. The FAS technique has been tried in various image processing applications; see e.g. [42,43,32,44,27,28,45,9,46,47]. We now denote our PDEs (5) by

\[
\begin{align*}
\mathcal{N}_1^h(u^h) &= g_1^h, \\
\mathcal{N}_2^h(u^h) &= g_2^h, \\
\end{align*}
\]

separating the linear operator \(L^h\) from the nonlinear operator \(f^h_l\) (\(l = 1, 2\)) on a general fine grid with step size \(h = (h_1, h_2)\) on \(\Omega_h\).

Let \(v^h = (v_1^h, v_2^h)^\top\) be the result computed by performing a few steps with a smoother (pre-smoothing step) on the fine-grid problem. Then, the algebraic error \(e^h\) of the solution is given by \(e^h = u^h - v^h\). The residual equation system is given by

\[
\begin{align*}
\mathcal{N}_1^h(v^h + e^h) - \mathcal{N}_1^h(v^h) &= g_1^h - \mathcal{N}_1^h(v^h) = r_1^h \\
\mathcal{N}_2^h(v^h + e^h) - \mathcal{N}_2^h(v^h) &= g_2^h - \mathcal{N}_2^h(v^h) = r_2^h.
\end{align*}
\]

In order to correct the approximated solution \(v^h\) on the fine grid, one needs to compute the error \(e^l\). However, the error cannot be computed directly. Since high-frequency components of the error in pre-smoothing step have already been removed by the smoother, we can transfer the following nonlinear system to the coarse grid as follows

\[
\begin{align*}
\mathcal{N}_1^H(v^H + e^H) &= r_1^H + \mathcal{N}_1^H(v^H) \\
\mathcal{N}_2^H(v^H + e^H) &= r_2^H + \mathcal{N}_2^H(v^H)
\end{align*}
\]

where \(H = 2h\) is the new cell size \(H_1 \times H_2\) with \(H_1 \geq h_1\) and \(H_2 \geq h_2\) and \(g_l^H \neq 0\) on the coarse grid. After the nonlinear residual equation system on the coarse grid (16) have been solved with a method of our choice, the coarse-grid correction \(e^H = u^H - v^H\) is then interpolated back to the fine grid \(e^h\) that can now be used for updating the approximated solution \(v^h\) on the fine grid, i.e. \(v^h_{\text{new}} = v^h + e^h\). The last step for a FAS multigrid is to perform the smoother again to remove high-frequency parts of the interpolated error (post-smoothing step).

In our FAS multigrid for diffusion image registration, standard coarsening is used in computing the coarse-grid domain \(\Omega_{H_l}\) by doubling the grid size in each space direction, i.e. \(h \rightarrow 2h = H\). For intergrid transfer operators between \(\Omega_h\) and \(\Omega_{H_l}\), the averaging and bilinear interpolation techniques are used for the restriction and interpolation operators denoted respectively by \(\pi_l^H\) and \(\iota_l^H\); see the details in [37,38,39,40]. In order to compute the coarse-grid operator of \(\mathcal{N}_l^H(u_l^H)\) consisting of two parts: \(L^H(u_l^H)\) and \(f_l^H(u_l^H, v_l^H)\), a so-called discretization coarse-grid approximation (DCA) is performed [37,31,30,40]. The idea is to rediscretize the Euler–Lagrange directly. In the case of \(L^H(u_l^H)\), the corresponding coarse-grid
part $\mathcal{L}^H(u^h_i)$ is obtained by the restriction of $u^h_i$ and a simple adaptation of the grid size to the discretized Laplacian. For $f_i^h(u^h_i, u^h_j)$, we first use the restriction operator with both components of the deformation field $u^h_i$, $u^h_j$, and the given images, $R^h$ and $T^h$, and then compute the corresponding coarse-grid part $f_i^H(u^h_i, u^h_j)$. To solve (14) numerically, our FAS multigrid is applied recursively down to the coarsest grid consisting of a small number of grid points, typically $4 \times 4$, and may be summarized as follows:

**Algorithm 1** (FAS Multigrid Algorithm). Denote FAS multigrid parameters as follows:

- $v_1$: pre-smoothing steps on each level
- $v_2$: post-smoothing steps on each level
- $l$: the number of multigrid cycles on each level ($l = 1$ for V-cycling and $l = 2$ for W-cycling). Here we present the V-cycle with $l = 1$.
- $\alpha$: regularization parameter
- $\omega$: relaxation parameter
- $\text{GSiter}$: the maximum number of iterations using a smoother

\[ [v_1^h, v_2^h] \leftarrow \text{FASMCY}(v_1^h, v_2^h, \rho^h_1, \rho^h_2, g^h_1, g^h_2, R^h, T^h, v_1, v_2, \alpha, \omega, \text{GSiter}). \]

For practical applications our FAS multigrid method is stopped if the maximum number of V- or W-cycles $\varepsilon_1$ is reached (usually $\varepsilon_1 = 20$), the mean of the relative residuals obtained from the Euler–Lagrangian equations is smaller than a small number $\varepsilon_2 > 0$ (typically $\varepsilon_2 = 10^{-8}$ for a convergence test and only $\varepsilon_2 = 10^{-1}$ for a practical registration), the relative reduction of the dissimilarity is smaller than some $\varepsilon_3 > 0$ (usually assign $\varepsilon_3 = 0.10$ meaning that the relative reduction of the dissimilarity would decrease about 90%), or the change in two consecutive steps of the data/fitting term $D$ is smaller than a small number $\varepsilon_4 > 0$ (typically $\varepsilon_4 = 10^{-8}$). A pseudo-code implementation of our FAS multigrid method is then summarized in the following algorithm:

**Algorithm 2** (FAS Multigrid Method).

\[ f^h \leftarrow \text{FASMG}(\alpha, \omega, v^h). \]

- Select $\alpha = (\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4)$ and initial guess solutions $v^h_{\text{initial}} = (v_1^h, v_2^h)^T$ on the finest grid.
- Set $K = 0$, $(v^h)^K = v^h_{\text{initial}}$, $\tilde{v}_2 = \varepsilon_2 + 1$, $\tilde{v}_3 = \varepsilon_3 + 1$, and $\tilde{v}_4 = \varepsilon_4 + 1$.
- While ($K < \varepsilon_1$ AND $\tilde{v}_2 > \varepsilon_2$ AND $\tilde{v}_3 > \varepsilon_3$ AND $\tilde{v}_4 > \varepsilon_4$)
  - \[ (v^h)^{K+1} = [v_1^h, v_2^h] \leftarrow \text{FASMCY}(v_1^h, v_2^h, \rho^h_1, \rho^h_2, g^h_1, g^h_2, R^h, T^h, v_1, v_2, \alpha, \omega, \text{GSiter}) \]
  - \[ \tilde{v}_2 = \text{mean}\left(\frac{|g^h_1 - g^h_2(v^h)^{K+1}|_2}{|g^h_1 - g^h_2(v^h)^{K}|_2}\right) \]
  - \[ \tilde{v}_3 = \text{mean}\left(\frac{|g^h_1 - g^h_2(v^h)^{K+1}|_2}{|g^h_1 - g^h_2(v^h)^{K}|_2}\right) \]
  - \[ \tilde{v}_4 = \text{mean}\left(\frac{|g^h_1 - g^h_2(v^h)^{K+1}|_2}{|g^h_1 - g^h_2(v^h)^{K}|_2}\right) \]
  - $K = K + 1$
- end
As is well known, in addition to restriction and interpolation operators, the above Algorithm 1 requires a suitable smoother based on some iterative relaxation method which is often the decisive factor in determining whether or not a multigrid algorithm converges. This issue is discussed next after we review the linear multigrid method.

2.4.2. The linear multigrid method for (5)

For a nonlinear problem, a linearization approach can lead to a coupled outer–inner iteration method with the inner solver provided by a linear multigrid method. For (5), the outer iteration is introduced either in a GN iteration [12,17,18] or in terms of the semi-implicit time-marching scheme [21] as follows

\[
\begin{align*}
(1 - \alpha \tau \sum_{l=1}^{2} A_l) u_1^{(k+1)} &= (u_1^{(k)} - \tau f_1(u_1^{(k)}, u_2^{(k)})), \\
(1 - \alpha \tau \sum_{l=1}^{2} A_l) u_2^{(k+1)} &= (u_2^{(k)} - \tau f_2(u_1^{(k)}, u_2^{(k)})),
\end{align*}
\]  

(17)

which is a system of two linear elliptic PDEs. Finally each outer step \( k \) is solved iteratively by a linear multigrid. In order to reduce the number of outer steps, a scale space framework described in [15] is used for adapting automatically the registration parameters \( \tau \) and \( \alpha \). Although, this numerical scheme is very accurate for providing visually pleasing registration results, we found experimentally that it is quite slow in fulfilling the necessary condition for being a minimizer of the variational problem represented by (2), i.e. in achieving convergence, because the linear system has to be solved many times with changing the right-hand side of (17). This is a convenient way of using a multigrid method but is not as optimal as a nonlinear multigrid method.

2.4.3. The nonlinear multigrid method for (5)

The above introduced FAS-NMG Algorithm 2 can be readily applied to (5). The choice of a suitable smoother is a key. Below we briefly review four types of smoothers that have been or can be used for diffusion image registration:

(1) **The Newton–Gauss–Seidel relaxation smoother.** This was used in [10] for (5). Although, there are no numerical results to demonstrate the convergence of the FAS technique, however, we found with several tests that this kind of smoothers provides visually pleasing registration results within a few multigrid steps. However, it does not perform well as a good smoother in leading to the convergence of the FAS-NMG method. Note that this smoother can be derived directly from (13).

(2) **The fixed point (FP) iteration based smoothers.** The simple linearized iterations by the following

\[
\begin{align*}
-\alpha \Delta u_1^{[v+1]} &= -(T(u^{[v]}) - R)\partial_{u_1} T(u^{[v]}) \\
-\alpha \Delta u_2^{[v+1]} &= -(T(u^{[v]}) - R)\partial_{u_2} T(u^{[v]})
\end{align*}
\]  

(18)

as discussed in [3, p. 79] have been used by researchers; we shall call this method the standard FP (SFP) below. This SFP scheme encounters a singular system for all fixed point step \( v \) due to Neumann’s boundary conditions. Without any special treatment for overcoming the singularity, we found that simple iterative methods such as the Jacobi-, GS-, or successive over-relaxation (SOR) type methods usually fail to lead to convergence of the FAS-NMG technique; however we discuss ways of improving this idea below.

(3) **The augmented system technique for the SFP scheme.** This was introduced in [8,9]. The idea is to convert the singular system to a nonsingular one by augmenting extra equations. However, it may not lead to satisfaction of the necessary (first-order) condition of a minimizer of the variational problem represented by (2). In details, this smoother builds on the above SFP method (18) with an initial guess \( u^{[0]} \)

\[
\begin{align*}
-\alpha \mathcal{L}(u_1^{[v+1]}) &= g_1 - f_1(u_1^{[v]}, u_2^{[v]}) = C_1^{[v]} \\
-\alpha \mathcal{L}(u_2^{[v+1]}) &= g_2 - f_2(u_1^{[v]}, u_2^{[v]}) = C_2^{[v]},
\end{align*}
\]  

(19)

Recall that \( g_1 = g_2 = 0 \) on the finest grid in multigrid setting and the symbol \( h \) and \((\cdot)_{ij}\) in (8) are dropped for simplicity. As already mentioned above, the resulting system is singular in case of Neumann’s boundary conditions approximated by (9). The reason for singularity is due to zero row sums at boundary points, i.e. the constant functions lie in the kernel of \( \mathcal{L} \). In this case the discrete system has a solution if and only if the discrete compatibility condition

\[
\sum_{j=1}^{n_1} C_1^{[v]} i_{ij} = 0 \quad \text{for} \quad l = 1, 2
\]

is satisfied [37,8,9,30]. Obviously, this condition fails when the given images are substantially different. Recognizing the above difficulties, Frohn-Schauf et al. [8,9] proposed to solve a nearby problem created by a simple modification, which guarantees that discrete solutions exist for each fixed point problem from replacing \( C_l^{[v]} (l = 1, 2) \) by
\[ \tilde{G}^v_l = G^v_l - \frac{(G^v_l, 1)}{(1, 1)} \frac{1}{\|1, 1\|}, \]

where \(1\) is the \(n_1 \times n_2\)-vector \((1, \ldots, 1)\). Note that if \(u^{[v+1]}\) solves the new discrete system

\[
\begin{aligned}
-\alpha \mathcal{L}(u_1^{[v+1]}) + c_1 u_1^{[v+1]} &= G_1^{[v]} + c_1 u_1^{[v]}, \\
-\alpha \mathcal{L}(u_2^{[v+1]}) + c_2 u_2^{[v+1]} &= G_2^{[v]} + c_2 u_2^{[v]}, \\
\end{aligned}
\]

\[ (MSFP-1) \tag{22} \]

then \(u^{[v+1]} + c\) also solves the same problem for any \(c\). This means that the solution is not unique. In order to determine the unique solution of the discrete system given by (20), they put a constraint on \(u^{[v+1]}\). This can be done by applying the zero-mean condition,

\[ \sum_{l,j=1}^{n_1,n_2} (u_l^{[v+1]})_{ij} = (u_l^{[v+1]}, 1) = c_l = 0 \quad \text{for} \quad l = 1, 2. \]

\[ (21) \]

We shall denote the above method by MSFP-FS (a modified SFP due to Frohn-Schauf et al.).

**4. The modified standard FP smoothers.** Following the same idea of overcoming singularities, below, consider 2 alternative ways of modifying the SFP method (18)

\[
\begin{aligned}
-\alpha \mathcal{L}(u_1^{[v+1]}) + c_1 u_1^{[v+1]} &= G_1^{[v]} + c_1 u_1^{[v]}, \\
-\alpha \mathcal{L}(u_2^{[v+1]}) + c_2 u_2^{[v+1]} &= G_2^{[v]} + c_2 u_2^{[v]}, \\
\end{aligned}
\]

\[ (MSFP-1) \tag{22} \]

\[
\begin{aligned}
(\alpha \mathcal{L} + \epsilon) u_1^{[v+1]} &= G_1^{[v]} \quad \text{(MSFP-2)} \tag{22a} \\
(\alpha \mathcal{L} + \epsilon) u_2^{[v+1]} &= G_2^{[v]}, \quad \text{(MSFP-2)} \tag{22b} \\
\end{aligned}
\]

where \(c_l \ (l = 1, 2)\) and \(\epsilon\) are positive real numbers (very small number for \(\epsilon\)). We note first that the modified SFP method of the first type (MSFP-1) given by (22) can be viewed as a semi-implicit time-marching scheme when \(c_1 = c_2 = c > 0\) is interpreted to be the time-step, as used in [3, p. 80]. Second, the second type (MSFP-2) represented by (23) is the simplest way to stabilize the SFP method by adding the small number \(\epsilon\) to all diagonal elements of \(\mathcal{L}\), as used in [43] in a different context. Finally, we note that determining the optimal values for the fixed point parameters \(c_1, c_2\) (or \(c\)), and \(\epsilon\) in automatic procedures leading to convergence of the FP method (and a multigrid technique) is not straightforward for real-world applications because tuning is needed for different registration problems.

In other tests, we found experimentally that although the approximation solutions obtained from the MSFP-FS scheme are visually pleasing, they may not fulfill the necessary condition for being a minimizer of the variational problem (2). The reason is that this numerical scheme solves a nearby problem for each fixed point step \(v\) by changing the right-hand side of (19) subject to the zero-mean condition (21).

These difficulties have motivated us to develop the new smoother in the next section.

3. The proposed algorithm with a new and robust smoother

To first design a better smoother for (5), we have to re-consider the nonlinear terms in (19) in a new FP scheme. Once this is done, a basic linear iterative solver such as the Jacobi, GS or SOR method for each corresponding system may be used. Then to improve the model robustness, we use a multi-resolution idea to choose the regularization parameter \(\alpha\).

3.1. The new FP smoother

Our idea of a new FP scheme is different from the SFP (18) and its variants, by aiming for full implicitness in both regularizer and data terms. This leads to

\[
\begin{aligned}
-\alpha \mathcal{L}(u_1^{[v+1]}) + f_1(u_1^{[v]}, u_2^{[v+1]}) &= g_1, \\
-\alpha \mathcal{L}(u_2^{[v+1]}) + f_2(u_1^{[v]}, u_2^{[v+1]}) &= g_2, \\
\end{aligned}
\]

\[ (24) \]

before we introduce linearization. We linearize the data term \(f_1(u_1^{[v+1]}, u_2^{[v+1]})\) via a first-order approximation of form (for \(l = 1, 2\))

\[
\begin{aligned}
f_l(u_1^{[v+1]}, u_2^{[v+1]}) &\sim f_l(u_1^{[v]}, u_2^{[v]}) + \delta_{u_l} f_l(u_1^{[v]}, u_2^{[v]}) \delta u_1^{[v]} + \delta_{u_2} f_l(u_1^{[v]}, u_2^{[v]}) \delta u_2^{[v]} \\
&= f_l(u_1^{[v]}, u_2^{[v]}) + \sigma_{1l}^{[v]} \delta u_1^{[v]} + \sigma_{2l}^{[v]} \delta u_2^{[v]}, \\
\end{aligned}
\]

\[ (25) \]

where

\[
\sigma_{1l}^{[v]} = \partial_{u_1} f_l(u_1^{[v]}, u_2^{[v]}) = (\partial_{u_1} T(u_1^{[v]}))(\partial_{u_1} T(u_1^{[v]})) + (T(u_1^{[v]}) - R)(\partial_{u_1} T(u_1^{[v]})),
\]

\[
\sigma_{2l}^{[v]} = \partial_{u_2} f_l(u_1^{[v]}, u_2^{[v]}) = (\partial_{u_2} T(u_1^{[v]}))(\partial_{u_2} T(u_1^{[v]})) + (T(u_1^{[v]}) - R)(\partial_{u_2} T(u_1^{[v]})),
\]
and

\[
\sigma_{12}(u^{[v]}_1) = \partial_{u_1} f(u^{[v]}_1, u^{[v]}_2) = (\partial_{u_1} T(u^{[v]}_1))(\partial_{u_2} T(u^{[v]}_2)) + (T(u^{[v]}_1) - R)(\partial_{u_2} T(u^{[v]}_2)).
\]

By (25), it leads to the semi-implicitly fixed point iteration step in terms of a \(2 \times 2\) matrix as follows

\[
\mathbf{N}[u^{[v]}_1]u^{[v+1]} = \mathbf{G}[u^{[v]}], \quad \text{(RFP)}
\]

where RFP refers to robust FP (yet to be tested shortly), \(u^{[v+1]} = (u^{[v+1]}_1, u^{[v+1]}_2)^\top\),

\[
\mathbf{N}[u^{[v]}] = \begin{bmatrix} -\alpha \mathbf{L} + \sigma_{11}(u^{[v]}_1) & \sigma_{12}(u^{[v]}_1) \\ \sigma_{21}(u^{[v]}_1) & -\alpha \mathbf{L} + \sigma_{22}(u^{[v]}_1) \end{bmatrix},
\]

\[
\mathbf{G}[u^{[v]}] = \begin{bmatrix} G_1^{[v]} + \sigma_{11}(u^{[v]}_1)u^{[v]}_1 + \sigma_{12}(u^{[v]}_1)u^{[v]}_2 \\ G_2^{[v]} + \sigma_{21}(u^{[v]}_1)u^{[v]}_1 + \sigma_{22}(u^{[v]}_1)u^{[v]}_2 \end{bmatrix}.
\]

To solve (26), we adopt the block or pointwise collective Gauss–Seidel (PCGS) relaxation method, i.e. all difference equations are updated simultaneously. A PCGS step is then given by

\[
(u^{[v+1]})^{[k+1]}_{ij} = (\mathbf{N}[u^{[v]}]_{ij})^{-1}(\mathbf{G}[u^{[v]}]_{ij})^{[k+1/2]},
\]

where using the same notation as (8)

\[
\mathbf{N}[u^{[v]}]_{ij} = \begin{bmatrix} (\alpha/h^2)\Sigma_{ij} + (\sigma_{11}(u^{[v]}_1))_{ij} & (\sigma_{12}(u^{[v]}_1))_{ij} \\ (\sigma_{21}(u^{[v]}_1))_{ij} & (\alpha/h^2)\Sigma_{ij} + (\sigma_{22}(u^{[v]}_1))_{ij} \end{bmatrix},
\]

\[
\mathbf{G}[u^{[v]}]^{[k+1/2]}_{ij} = \begin{bmatrix} (c_1^{[v]}_{ij})_{ij} + (\sigma_{11}(u^{[v]}_1))_{ij}u^{[v]}_1 + (\sigma_{12}(u^{[v]}_1))_{ij}u^{[v]}_2 + (\alpha/h^2)\Sigma_{ij}u^{[v+1]}_1^{[k+1/2]}_{ij} \\ (c_2^{[v]}_{ij})_{ij} + (\sigma_{21}(u^{[v]}_1))_{ij}u^{[v]}_1 + (\sigma_{22}(u^{[v]}_1))_{ij}u^{[v]}_2 + (\alpha/h^2)\Sigma_{ij}u^{[v+1]}_2^{[k+1/2]}_{ij} \end{bmatrix},
\]

\[
(\Sigma)_{ij}(u^{[v+1]}_1)^{[k+1/2]}_{ij} = (u^{[v+1]}_1)^{[k+1]}_{j} + (u^{[v+1]}_2)^{[k+1]}_{j} + \gamma^2 (u^{[v+1]}_1)^{[k]}_{j} + \gamma^2 (u^{[v+1]}_2)^{[k]}_{j}.
\]

To gain more efficiency, one may introduce a relaxation parameter \(\omega \in (0, 2)\) and iterate the \(\omega\)-PCGS steps by

\[
(u^{[v+1]})^{[k]}_{ij} = (1 - \omega)(u^{[v+1]}_1)^{[k]}_{ij} + \omega (\mathbf{N}[u^{[v]}]_{ij})^{-1}(\mathbf{G}[u^{[v]}]_{ij})^{[k+1/2]}.
\]

We note first that the proposed smoother given by (26) shows the interaction between the actual fixed point iteration that overcomes the nonlinearity of the operator \(\mathcal{L}\) at each outer step \(v\) and the \(\omega\)-PCGS method that solves the resulting linear system of equations at each corresponding inner step \(k\). Second, instead of solving the linear system of equations using the inner solver (28) with very high precision, it can perform only a few iteration to obtain an approximation solution at each outer step. Evidently, this procedure leads to a slight difference of convergence in the fixed point scheme when the proposed smoother is used as a stand-alone solver, whereas the computational costs significantly reduce; see Fig. 1(a). Moreover, the relaxation parameter \(\omega\) also has a strong influence on the convergence speed. As the stand-alone solver of (26) we usually use \(\omega > 1\), typically \(\omega = 1.85\), because it results in speeding up the convergence compared with the PCGS approach (\(\omega = 1\); see Fig. 1(b). For our multigrid algorithm, however, we use the local Fourier analysis and several experiments to select the optimal value of \(\omega\); see Section 3.2 later. Finally we remark that other iterative techniques such as the line relaxation techniques or the preconditioned conjugate gradient method may also be used as an inner solver. However, the \(\omega\)-PCGS relaxation method appears a cheaper option for practical applications.

Implementation of our proposed smoother (26) based on the \(\omega\)-PCGS method (28) on a fine grid can be summarized as follows:

**Algorithm 3 (Our Proposed Smoother).** Denote by

- \(\alpha\) regularization parameter
- \(\omega\) relaxation parameter
- \(\text{GSite}\) the maximum number of \(\omega\)-PCGS iterations

\[
[v_1^h, v_2^h] \leftarrow \text{Smoother}(v_1^h, v_2^h, g_1^h, g_2^h, R^h, T^h, \alpha, \omega, \text{GSite}).
\]
Operator can be linearized (by freezing coefficients) and replaced locally by a new operator with constant coefficients.

Grid is assumed to eliminate the effect of boundary conditions and second it is also assumed that the discrete nonlinear equation (26) is used to derive the FP schemes. We also wish to remark that the above quantities $\sigma_{i1}(u^{[v]}), \sigma_{i2}(u^{[v]})$ may be refined to derive a cheaper implementation than our FP method (26). We note first that $\sigma_{21}(u^{[v]}) = \sigma_{12}(u^{[v]})$. Second in order to have a simple and stable numerical scheme as pointed out by several works; see e.g. [18] and [3, p. 56/79], we approximate $\sigma_{lm}(u^{[v]})$ by $\sigma_{lm}(u^{[v]}) = (\partial_{u_l} T(u^{[v]}))(\partial_{u_m} T(u^{[v]}))$ for $m = 1, 2$ since the image difference $T(u^{[v]}) - R$ becomes small for well registered images and the second-order derivatives of $T(u^{[v]})$ ($\partial_{u_m} T(u^{[v]})$), a problematic part of $\sigma_{lm}(u^{[v]})$, are very sensitive to noise and are hard to estimate robustly. Finally, we note that if discrete image gradient $\partial_{u_l} T(u^{[v]})$ does not vanish at one point, the system matrix of these linearized equations is strictly or irreducibly diagonally dominant. This guarantees the existence of a unique solution of each linearized system and global convergence of the Jacobi and GS iterations [49,50].

Below we analyze the smoothing property of our proposed smoother (26).

### 3.2. Local Fourier analysis (LFA)

LFA is a powerful tool to analyze the smoothing properties of iterative algorithms used in MG methods. Although LFA was originally developed for discrete linear operators with constant coefficients on infinite grids, it can also be applied to more general nonlinear equations with varying coefficients such as the discrete version of (5). To this end, first an infinite grid is assumed to eliminate the effect of boundary conditions and second it is also assumed that the discrete nonlinear operator can be linearized (by freezing coefficients) and replaced locally by a new operator with constant coefficients [30].
This approach has proved to be very useful in the understanding of MG methods when solving nonlinear problems; see for instance [42, 51, 43, 44, 45, 12, 53, 54, 18, 55] for interesting examples and discussions.

3.2.1. Measure of $h$-ellipticity

It is well known that $h$-ellipticity is crucial for multigrid methods to be effective. It is often used to decide whether or not pointwise error smoothing procedures (e.g. our proposed smoother (26) based on (28)) can be constructed for the discrete operator under consideration. To this end, we shall show that the linearized system $N_h[\mathbf{u}^h] = G_h[\mathbf{u}^h]$ in (26) at some outer step provides a sufficient amount of $h$-ellipticity in a similar way shown in [12, 18, 30, 40] for a discrete system of PDEs. Here $\mathbf{u}^h$ and $\mathbf{u}^h$ denote the exact solution and the current approximation and $N_h[\mathbf{u}^h]$ and $G_h[\mathbf{u}^h]$ the resulting discrete operators from the linearization at $\mathbf{u}^h$. For simplicity, our analysis is carried out over the infinite grid

$$\Omega^\infty_h = \{x \in \Omega | x = (x_i, x_j)^T = ((2i - 1)/h, (2j - 1)/h)^T, \ i, j \in \mathbb{Z}^2\}$$

(30)

where $h = 1/n$ denotes the mesh parameter.

Let $\varphi_h(\theta, x) = \exp(i \theta x/h) \cdot 1$ be grid functions, where $\hat{\theta} = (1, 1)^T$, $\theta = (\theta_1, \theta_2)^T \in \Theta = (-\pi, \pi)^2$, $x \in \Omega^\infty_h$, and $i = \sqrt{-1}$. It is important to remark that due to the locality nature of LFA, our analysis applies to each grid point separately, i.e. we consider the local discrete system $N_h(\xi) \mathbf{u}^h = G_h(\xi)$ centered and defined only within a small neighborhood of each grid point $\xi = (i, j)$ and $\mathbf{u}^h(\xi) = \{u^h_i(\xi), u^h_j(\xi)\}$. Applying the discrete operator $N_h(\xi)$ to the grid functions $\varphi_h(\theta, x)$, i.e. $N_h(\xi) \varphi_h(\theta, x) = N_h(\xi, \theta) \varphi_h(\theta, x)$, yields the Fourier symbol as follows:

$$\widehat{N}_h(\xi, \theta) = \begin{bmatrix} -\alpha \hat{L}^h(\theta) + \sigma_{11}(\xi) \\
\sigma_{11}(\xi) \\
\sigma_{11}(\xi) \end{bmatrix}$$

(31)

(see details related to Fourier symbols of systems of PDEs in [30, 40]). Here $\hat{L}^h(\theta)$ denotes the Fourier symbol of the discrete Laplacian operator $\hat{L}^h$. Following [30, 40], the measure of $h$-ellipticity is defined via $\widehat{N}_h(\xi, \theta)$ as follows:

$$E_h(\widehat{N}_h(\xi, \theta)) = \frac{\min\{|\det(\widehat{N}_h(\xi, \theta))| : \theta \in \Theta_{\text{high}}\}}{\max\{|\det(\widehat{N}_h(\xi, \theta))| : \theta \in \Theta\}}.$$

(32)

where $\Theta_{\text{high}} = \Theta \setminus (-\pi/2, \pi/2)^2$ denotes the range of high frequencies and

$$\det(\widehat{N}_h(\xi, \theta)) = \alpha^2(\hat{L}^h(\theta))^2 + \alpha c_1(\hat{L}^h(\theta)) + c_2$$

(33)

represents the determinant of $\widehat{N}_h(\xi, \theta)$ where

$$c_1 = -(\sigma_{11}(\xi) + \sigma_{22}(\xi)) \quad \text{and} \quad c_2 = \sigma_{11}(\xi)\sigma_{22}(\xi) - \sigma_{12}(\xi)\sigma_{21}(\xi).$$

For the Laplacian, we have

$$-\hat{L}^h(\theta) = (2/h^2)(2 - (\cos \theta_1 + \cos \theta_2)), \quad \min_{\theta \in \Theta_{\text{high}}} (-\hat{L}^h(\theta)) = -\hat{L}^h(-\pi/2, 0) = 2/h^2$$

and $\max_{\theta \in \Theta} (-\hat{L}^h(\theta)) = -\hat{L}^h(\pi, \pi) = 8/h^2$. Therefore,

$$E_h(\widehat{N}_h(\xi, \theta)) = \frac{2\alpha + c_1 h^2 + (c_2 h^4/2\alpha)}{32\alpha + 4c_1 h^2 + (c_2 h^4/2\alpha)}$$

(34)

and

$$\lim_{h \to 0} E_h(\widehat{N}_h(\xi, \theta)) = \frac{1}{16}$$

(35)

bounded away from 0 for all possible choices $\alpha, h > 0$ and for all possible values of $\sigma_{11}(\xi), \sigma_{12}(\xi), \sigma_{21}(\xi)$, and $\sigma_{22}(\xi)$ (i.e. the results do not depend on the given images) over the whole discrete domain $\Omega^h$.

As a result, the discrete system $N_h[\mathbf{u}^h] = G_h[\mathbf{u}^h]$ is appropriate to pointwise error smoothing procedures and so is our proposed smoother (26) combined with the $\omega$-PCGS method (28).

3.2.2. Smoothing analysis for the proposed smoother

A robust and optimal smoother has to take care of the high-frequency components of the error between the exact solution and the current approximation since the low-frequency components becomes the high-frequency components on coarser grids and they cannot be reduced on coarser grids by the coarse-grid correction procedure. A quantitative measure of the smoothing efficiency for a given algorithm is the smoothing factor denoted by $\mu$ from a LFA and numerically computed for test problems, which is defined as the worst asymptotic error reduction, by performing one smoother step, of all high-frequency error components [30, 40]. Below we shall analyze the smoothing properties of the proposed smoother via (26) and (28).
As pointed out in many cases of nonlinear operators with varying coefficients in [42,51,43,44,52,12,53,54,18,55], the smoothing factor is \( x \)-dependent. Therefore, it is customary to look for the maximum over the local smoothing factors of the frozen operator \( N_h(\xi) \), i.e.
\[
\mu_{\text{loc}} = \max_{\xi \in \Omega_h} \mu_{\text{loc}}.
\]
To determine \( \mu_{\text{loc}} \) we consider again the local discrete system \( N_h(\xi)u^h(\xi) = G_h(\xi) \). By using the splitting \( N_h(\xi) = N^{+1}_h(\xi) + N^{0}_h(\xi) + N^{-1}_h(\xi) \), it is possible to write the local inner iterations of (26) (for \( \omega = 1 \)) as
\[
N^{+1}_h(\xi)u_{\text{new}}^h(\xi) + N^0_h(\xi)u_{\text{old}}^h(\xi) + N^{-1}_h(\xi)u_{\text{old}}^h(\xi) = G_h(\xi)
\]
where \( u_{\text{old}}^h(\xi) \) and \( u_{\text{new}}^h(\xi) \) stand for the approximations to \( u^h(\xi) \) before and after the inner smoothing step, respectively and
\[
N^{+/0/-1}_h(\xi) = \begin{pmatrix}
N^{+/0/-1}_h(\xi)_{1,1} & N^{+/0/-1}_h(\xi)_{1,2} \\
N^{+/0/-1}_h(\xi)_{2,1} & N^{+/0/-1}_h(\xi)_{2,2}
\end{pmatrix}.
\]
For a specification of this splitting, we use the stencil notation as follows:
\[
L^h_{l+1} = \frac{\alpha}{h^2} \begin{pmatrix}
0 & 0 & 0 \\
-1 & 0 & 0 \\
0 & 0 & 0
\end{pmatrix}, \quad L^h_0 = \frac{\alpha}{h^2} \begin{pmatrix}
0 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 0
\end{pmatrix}, \quad L^h_{l-1} = \frac{\alpha}{h^2} \begin{pmatrix}
0 & -1 & 0 \\
0 & 0 & -1 \\
0 & 0 & 0
\end{pmatrix},
\]
and
\[
\kappa = \begin{bmatrix}
\kappa_{1,1} & \kappa_{1,2} \\
\kappa_{2,1} & \kappa_{2,2}
\end{bmatrix} = \begin{bmatrix}
\Sigma + (h^2/\alpha)\sigma_{11}(\xi) & (h^2/\alpha)\sigma_{12}(\xi) \\
(h^2/\alpha)\sigma_{21}(\xi) & \Sigma + (h^2/\alpha)\sigma_{22}(\xi)
\end{bmatrix}.
\]
By subtracting (37) from \( N_h(\xi)u^h(\xi) = G_h(\xi) \) and defining \( \tilde{e}_h^b(\xi) = u^h(\xi) - u_{\text{new}}^h(\xi) \) and \( \tilde{e}_h^o(\xi) = u^h(\xi) - u^o(\xi) \) we obtain the local system of error equations
\[
N^{+1}_h(\xi)\tilde{e}_{\text{new}}^h(\xi) + N^0_h(\xi)\tilde{e}_{\text{new}}^h(\xi) + N^{-1}_h(\xi)\tilde{e}_{\text{old}}^h(\xi) = 0
\]
or
\[
\tilde{e}_{\text{new}}^h(\xi) = -[N^0_h(\xi) + N^{+1}_h(\xi)]^{-1}[N^{-1}_h(\xi)\tilde{e}_{\text{old}}^h(\xi)] = S_h(\xi)\tilde{e}_{\text{old}}^h(\xi)
\]
where \( S_h(\xi) \) is the amplification factor. The effect of \( S_h(\xi) \) on the grid functions \( \varphi_h(\theta, x) \) within \( \Theta_{\text{high}} = \Theta \setminus [-\pi/2, \pi/2]^2 \) will determine the smoothing properties of the PCGS method (27).
For the \( \omega \)-PCGS approach (28), the amplification factor denoted by \( S_h(\xi, \omega) \) can be defined in a similar way as (38) and its Fourier symbol is given by
\[
\hat{S}_h(\xi, \theta, \omega) = \hat{N}_h^{\omega}(\xi, \theta) + \omega \hat{N}_h^{\omega}(\xi, \theta)\hat{N}_h^{\omega}(\xi, \theta) = S_h(\xi)\hat{e}_h^b(\xi)
\]
where \( \Theta_{\text{high}} \) is the high resolution. Here the Fourier symbols of \( N^{+/0/-1}_h(\xi) \) are
\[
\hat{N}^{+/0/-1}_h(\xi, \theta) = \begin{pmatrix}
\hat{N}^{+/0/-1}_h(\xi, \theta)_{1,1} & \hat{N}^{+/0/-1}_h(\xi, \theta)_{1,2} \\
\hat{N}^{+/0/-1}_h(\xi, \theta)_{2,1} & \hat{N}^{+/0/-1}_h(\xi, \theta)_{2,2}
\end{pmatrix}
\]
Therefore, the local smoothing factor is
\[
\mu_{\text{loc}} = \sup\{|\rho(\hat{S}_h(\xi, \theta, \omega))| : \theta \in \Theta_{\text{high}}\}
\]
where \( \rho \) indicates the spectral radius of \( \hat{S}_h(\xi, \theta, \omega) \). Recall that
\[
\begin{align*}
(N^{+1}_h(\xi, \theta))_{l,m} &= \begin{cases}
-\frac{\alpha}{h^2}(\exp(-i\theta_1) + \exp(-i\theta_2)) & \text{for } l = m \\
0 & \text{for } l \neq m,
\end{cases} \\
(N^{-1}_h(\xi, \theta))_{l,m} &= \begin{cases}
-\frac{\alpha}{h^2}(\exp(i\theta_1) + \exp(i\theta_2)) & \text{for } l = m \\
0 & \text{for } l \neq m
\end{cases} \\
(N^0_h(\xi, \theta))_{l,m} &= \frac{\alpha}{h^2}\kappa_{l,m}
\end{align*}
\]
will be used to compute (39).
To select the optimal value of \( \omega \) and test our smoother we consider one set of medical images as shown respectively in Fig. 3(a)-(b) on a \( 32 \times 32 \) grid. Fig. 2 shows the smoothing factors of the proposed smoother (26) based on the \( \omega \)-PCGS approach (28) at different values of \( \omega \). It indicates that the optimal value \( \omega \) providing \( \mu_{\text{loc}}^* \approx 0.5 \) is not exactly 1 but very close to 1, typically \( \omega \approx 0.9725 \).
3.3. A nonlinear multigrid algorithm with automatic selection of $\alpha$

As is typical of Tikhonov regularization, the energy functional $J_\alpha$ in (2) has a regularization parameter $\alpha$. To provide well matched images, we have to carefully select $\alpha$ because it is in general unknown a priori. In order to find a suitable $\alpha$ automatically, we follow the ‘cooling’ (‘continuation’) process suggested in [56–58,12,15,59,60]. The basic idea is to start with a high initial value of $\alpha$ and then slowly reduce $\alpha$ such that the obtained solution can be used to be an excellent starting point for the next in order to decrease $J_\alpha$. An alternative approach can be the L-curve method.

Consider the discrete version of the minimization problem (2) with the same notation

$$\min J_\alpha[u] = D(R, T(u)) + \alpha R(u). \quad (45)$$

Let $\alpha_1$ be the initial value, which is sufficiently large. At the $(s+1)$th step we set $\alpha^{(s+1)} = \beta \alpha^{(s)} \in [\alpha_0, \alpha_1]$, where $\beta \in (0, 1)$ is a constant, usually chosen to be about 0.5, and $\alpha_0$ is a small positive number, e.g. $5 \times 10^{-5}$. Subsequently, we apply $\alpha^{(s+1)}$ and the initial guess solution obtained by the previous iteration $u^{(s+1)}_{\text{init}} = u^{(s)}$ with the associated inner loop to obtain the minimum $u^{(s+1)}$ within some tolerance. As mentioned in [58], since the functional $J_\alpha$ is changing at each outer loop iteration, the demand of decreasing the value of the same functional is not reasonable. Then, the solution $u^{(s+1)}$ and parameter $\alpha^{(s+1)}$ are acceptable if they satisfy the so-called consistent condition:

$$J_\alpha[u^{(s+1)}] = D[u^{(s+1)}] + \alpha^{(s+1)} R[u^{(s+1)}] < J_\alpha[u^{(s)}] = D[u^{(s)}] + \alpha^{(s)} R[u^{(s)}].$$

However, if this condition is not satisfied, we increase $\beta$ (usually to 0.9) and re-start the step. Our experience suggests that the stopping criterion given by

$$\frac{||u^{(s+1)} - u^{(s)}||_2}{\max(||u^{(s+1)}||_2, ||u^{(s)}||_2)} < \delta$$

is suitable, where $\delta > 0$ is small (normally set to $10^{-3}$).

Finally, we summarize this process as follows:

**Algorithm 4 (Multigrid Image Registration Through Cooling).**

$[\mathbf{v}^*, \alpha^*] \leftarrow$ cooling$(v^{(0)}, \alpha^{(0)}, \mathbf{e})$.

- Set $s = 1$, $v^{(s)} = v^{(0)}$, $\alpha^{(s)} = \alpha^{(0)}$, $\beta = 0.5$.

- **Outer iteration:** For $s = 1, 2, 3, \ldots$
  - 1. Set $\alpha^{(s+1)} = \beta \alpha^{(s)}$ in $[5 \times 10^{-5}, \alpha^{(s)}]$.
  - 2. Inner iteration: $v^{(s)} \leftarrow$ FASMG$(v^{(s)}, \alpha^{(s+1)}, \mathbf{e})$.
  - 3. If $J_\alpha[v^{(s)}] < J_\alpha[v^{(s+1)}]$, go to 4.
  - 3.1 Set $v^{(s+1)} = v^{(s)}$, $\beta = 0.5$, $s = s + 1$, and go to 4.
    - Else
      - 3.2 Set $\beta = 0.9$, and go to 4.
  - 4. Check for convergence using the criterion (46).
    - If not satisfied, then return to 1, else, exit to the next step to stop.
  - Set $\mathbf{v}^* = v_{\text{new}}$ and $\alpha^* = \alpha^{(s)}$. 

![Fig. 2. Smoothing factors $\mu_{\text{loc}}^*$ at a fixed value of $\alpha = 0.1$ after 5 outer iterations with GSiter = 5 by the proposed smoother (26) based on the $\omega$-PGS approach (28) with different values of $\omega$ for the registration problem in Examples 1 as shown in Fig. 3(a)–(b) on a 32 x 32 grid. The red diamond indicates the optimal value of $\omega$.](image-url)
In order to save computational work for high-resolution digital images, the low-tolerance \( \overrightarrow{\mathbf{e}}_{l_0} = (2, 10^{-4}, 0.1, 10^{-4}) \) is applied to reduce the accumulated costs in each minimization problem. Then our first algorithm, namely a robust diffusion image registration (RDR) approach, can be stated as follows:

**Algorithm 5 (The Basic RDR Method).**

1. Input \( \overrightarrow{\mathbf{e}}_{l_0} \). Set \( \alpha = 1 \) (optional). Set \( \overrightarrow{\mathbf{e}}_{l_0} = (20, 10^{-8}, 0.10, 10^{-8}) \) (high-tolerance).
2. Obtain the optimal regularization parameter \( \alpha \) (through cooling) via Algorithm 4:
   \( - \mathbf{v}^{(0)}, \alpha \leftarrow \text{cooling} (\mathbf{v}, \alpha, \overrightarrow{\mathbf{e}}_{l_0}). \)
3. Solve the discrete minimization problem (45) on the finest level using the found \( \alpha \):
   \( - \mathbf{v} \leftarrow \text{FASMG}(\mathbf{v}^{(0)}, \alpha, \overrightarrow{\mathbf{e}}_{l_0}). \)

Although the above algorithm enables us to find a good \( \alpha \), the cost of resolving the same problem repeatedly is expensive. We propose to use a hierarchy of \( L \) grids (with level \( L \) the finest and level 1 the coarsest one) using a multi-resolution idea to gain efficiency while finding an effective \( \alpha \). Firstly we shall seek the optimal \( \alpha \) on the coarsest level 1 with the grid size of \( 32 \times 32 \) only (believed to be coarse enough) and secondly we use the multilevel continuation idea [12] to provide the initial guesses for the next finer level.

**Algorithm 6 (Multilevel Grid Continuation for Optimal \( \alpha \) and Reliable Initial Solution).**

\[
[\mathbf{v}^{[\text{lev}]}, \alpha^{[\text{lev}]}] \leftarrow \text{RDR\_multiresolution}(\mathbf{v}^{[\text{lev}]}, \alpha^{[\text{lev}]}, \text{lev}, \overrightarrow{\mathbf{e}}).
\]

- If \( \text{lev} = 1 \):
  \( - \mathbf{v}^{[\text{lev}]} = 0 \)
  \( - \alpha^{[\text{lev}]}, C > 0 \) should be large enough e.g.
  \( C = 100 \)
  \( - [\mathbf{v}^{[\text{lev}]}, \alpha^{[\text{lev}]}] \leftarrow \text{cooling} (\mathbf{v}^{[\text{lev}]}, \alpha^{[\text{lev}]}, \overrightarrow{\mathbf{e}}) \)
- Else
  \( - \mathbf{v}^{[\text{lev}]} = ((H_1^H, v_1^{[\text{lev}]}, H_1^H v_2^{[\text{lev}]}) \top \)
  \( - [\mathbf{v}^{[\text{lev}]}], \alpha^{[\text{lev}]}] \leftarrow \text{RDR\_multiresolution}(\mathbf{v}^{[\text{lev}]-1}, \alpha^{[\text{lev}]-1}, \text{lev} - 1, \overrightarrow{\mathbf{e}}) \)
  \( - \mathbf{v}^{[\text{lev}]} = ((H_1^H, v_1^{[\text{lev}]-1}, H_1^H v_2^{[\text{lev}]-1}) \top \)
  \( - \alpha^{[\text{lev}]} = 4 \alpha^{[\text{lev}]-1} \) [Recall that \( \alpha^{[\text{lev}]} = \overrightarrow{\mathbf{e}} n_{\text{lev}}^2 \) and \( n_{\text{lev}} = 2 n_{\text{lev}+1} \)]
  \( - \mathbf{v}^{[\text{lev}]} \leftarrow \text{FASMG}(\mathbf{v}^{[\text{lev}]}, \alpha^{[\text{lev}]}, \overrightarrow{\mathbf{e}}) \)
- Endif

Finally the overall procedure of finding an optimal \( \alpha \) and then starting a nonlinear multigrid method to solve (2) is summarized below as Algorithm 7:

**Algorithm 7 (The Refined RDR Multi-Resolution Method).**

1. Input \( \overrightarrow{\mathbf{e}}_{l_0} \) and \( \overrightarrow{\mathbf{e}}_{l_0} \).
2. Obtain the optimal regularization parameter \( \alpha \) (through cooling) and a good initial solution (through multi-resolution) \( \mathbf{v}^{(0)} \) via Algorithm 6:
   \( - [\mathbf{v}^{(0)}, \alpha] \leftarrow \text{RDR\_multiresolution}(\mathbf{v}^{[1]}, \alpha^{[1]}, L, \overrightarrow{\mathbf{e}}_{l_0}). \)
3. Solve the minimization problem (45) on the finest level \( \text{lev} = L \) using the found \( \alpha \) and the initial guess solution \( \mathbf{v}^{(0)} \):
   \( - \mathbf{v}^{[\text{lev}]} \leftarrow \text{FASMG}(\mathbf{v}^{(0)}, \alpha, \overrightarrow{\mathbf{e}}_{l_0}). \)
4. An application of Algorithm 7 to a curvature model

To test the robustness of our numerical algorithm for other registration models, in this section, we shall examine our second test model, namely, the curvature image registration model, as introduced in [5,6]; see also [7,18,33,3].

The curvature model. Based on an approximation of the mean curvature of the surface of \( u_i \), Fischer–Modersitzki’s curvature approach aims to find a reasonable deformation field \( u \) that minimizes the following functional [5,6]

\[
\mathcal{J}_a(u) = \mathcal{D}(u) + \alpha \mathcal{R}^C(u)
\]

where \( \mathcal{R}^C \) is as in (4). This leads to the Euler–Lagrange system of two fourth-order nonlinear PDEs:

\[
\begin{align*}
    f_1(u) + \alpha \Delta^2 u_1 &= 0, \\
    f_2(u) + \alpha \Delta^2 u_2 &= 0
\end{align*}
\]

(47)

subject to the special boundary conditions \( \nabla u_l = 0, \nabla \Delta u_l = 0 \) on \( \partial \Omega \), for \( l = 1, 2 \). We remark that the use of second-order derivatives in the energy functional (4) not only provides smoother deformation fields \( u \) than those of (3), but also allows for an automatic rigid alignment. Here \( u_i \) is understood as a surface in \( \mathbb{R}^3 \) represented by \((x_1, x_2, u_i(x_1, x_2))\), where initially \( u_i(x_1, x_2) = 0 \), with the mean curvature of the surface of \( u_i \) is given by

\[
\kappa_M(u_i) = \nabla \cdot \frac{\nabla u_i}{\sqrt{1 + |\nabla u_i|^2}} = \frac{(1 + u_i^2) u_{i_1_1} - 2u_{i_1} u_{i_2} u_{i_2_2} + (1 + u_i^2) u_{i_2_2}}{(1 + u_i^2 + u_i^4)^{3/2}}.
\]

(48)

Observe that \( |\nabla u_i| \approx 0 \) yields \( \kappa_M(u_i) \approx \hat{\kappa}_M(u_i) = \Delta u_i \). See also [24].

Since the biharmonic operator which appears in (47) is known to lead to poor multigrid efficiency, therefore, it is common to split up the equation into a system of two Poisson-type equations; see [37,38,16,18,30,39,40]. Based on this splitting idea, (47) can be converted to the following system using additional unknown functions \( v_1 = -\Delta u_1 \) and \( v_2 = -\Delta u_2 \):

\[
\begin{align*}
    -\Delta u_1 - v_1 &= 0, \\
    -\Delta u_2 - v_2 &= 0, \\
    f_1(u) - \alpha \Delta v_1 &= 0, \\
    f_2(u) - \alpha \Delta v_2 &= 0
\end{align*}
\]

(49)

subject to the boundary conditions transferred into \( \nabla u_l = 0 \) and \( \nabla v_l = 0 \) on \( \partial \Omega \) where the data term \( f_i(u) \) is given by (5).

To solve the above continuous system numerically, (49) is first discretized by the cell-centered finite difference scheme over the discrete domain

\[
\Omega_h = \{ \mathbf{x} \in \Omega | \mathbf{x} = (x_1, x_2)^T = ((2i - 1)h/2, (2j - 1)h/2), \ 1 \leq i, j \leq n \}
\]

consisting of \( N = n^2 \) cells of size \( h \times h \) with grid spacing \( h = 1/n \). Let \((z^h_i)_{i,j} = z^h_i(x_1, x_2)\) denote the grid function for \( \hat{l} = 1, \ldots, 4 \), where \( z^h_i = u^h_i \) or \( v^h_i \) for \( l = 1, 2 \). Then, the discrete system of (49) at a grid point \((i, j)\) is given by

\[
\begin{align*}
    -z^h_i(u^h_1)_{i,j} - (v^h_1)_{i,j} &= (g^h_1)_{i,j}, \\
    -z^h_i(u^h_2)_{i,j} - (v^h_2)_{i,j} &= (g^h_2)_{i,j}, \\
    f_1^h(u^h_1, u^h_2)_{i,j} - \alpha L^h(v^h_1)_{i,j} &= (g^h_3)_{i,j}, \\
    f_2^h(u^h_1, u^h_2)_{i,j} - \alpha L^h(v^h_2)_{i,j} &= (g^h_4)_{i,j},
\end{align*}
\]

(50)

where \((z^h)_{i,j} = ((z^h_1)_{i,j}, (z^h_2)_{i,j}, (z^h_3)_{i,j}, (z^h_4)_{i,j})^T = ((u^h_1)_{i,j}, (u^h_2)_{i,j}, (v^h_1)_{i,j}, (v^h_2)_{i,j})^T \) and \((g^h)_{i,j} = 0 \) \((\hat{l} = 1, \ldots, 4)\) on the finest grid in multigrid setting. Recall that the discrete versions of \( L^h \) and \( f_i^h(u^h_1, u^h_2)_{i,j} \) are given in the same way as represented by Section 2.2 and the approximations used in (50) need to be modified at grid points near the image boundary \( \partial \Omega_h \) using the homogeneous Neumann boundary conditions approximated by one-side differences for boundary derivatives:

\[
(z^h_{i+1})_1 = (z^h_i)_2, \quad (z^h_{i+1})_n = (z^h_i)_{n-1}, \quad (z^h_{i+1})_1 = (z^h_i)_2, \quad (z^h_{i+1})_n = (z^h_i)_{n-1}.
\]

(51)

A robust smoother based on the linearization scheme (25). As mentioned above, our aim is to apply our linearization idea (25) in solving the equivalent system (49). This leads to the linearized system

\[
N^{C^v}[z^v]z^{v+1} = G^v[z^v].
\]

(52)
where the symbols \( h \) and \((\cdot)_{i,j} \) in (50) are dropped for simplicity. Here
\[
N^\psi [z^{[v]}] = \begin{bmatrix} -\mathcal{L} & 0 & -1 & 0 \\ 0 & -\mathcal{L} & 0 & -1 \\ \sigma_{11}(u^{[v]}) & \sigma_{12}(u^{[v]}) & -\alpha\mathcal{L} & 0 \\ \sigma_{21}(u^{[v]}) & \sigma_{22}(u^{[v]}) & 0 & -\alpha\mathcal{L} \end{bmatrix}
\] (53)
and
\[
G^\psi [z^{[v]}] = \begin{pmatrix} g_1 \\ g_2 \\ g_3 - f_1(u^{[v]}, u^{[v]}) + \sigma_{11}(u^{[v]})u_1^{[v]} + \sigma_{12}(u^{[v]})u_2^{[v]} \\ g_4 - f_2(u_1^{[v]}, u_2^{[v]}) + \sigma_{21}(u^{[v]})u_1^{[v]} + \sigma_{22}(u^{[v]})u_2^{[v]} \end{pmatrix}.
\] (54)
As mentioned in Section 3.1, \( \sigma_{21} = \sigma_{12} \) and \( \sigma_{lm} = (\partial_{u_l} T(u^{[v]}))(\partial_{u_m} T(u^{[v]})) \) for \( m = 1, 2 \) are used to stabilize our numerical scheme. In order to solve (52), we again apply the \( \omega \)-PCGS as the inner solver and its kth step is updated by
\[
(z^{[v] + 1})_{i,j}^{[k]} = (1 - \omega)(z^{[v] + 1})_{i,j}^{[k]} + \omega(N^\psi [z^{[v]}]_{i,j} - 1)(G^\psi [z^{[v]}]_{i,j}^{[k+1/2]})
\] (55)
where
\[
N^\psi [z^{[v]}]_{i,j} = \begin{pmatrix} (\Sigma)_{i,j}/h^2 & 0 & -1 & 0 \\ 0 & (\Sigma)_{i,j}/h^2 & 0 & -1 \\ (\sigma_{11}(u^{[v]})_{i,j}) & (\sigma_{12}(u^{[v]})_{i,j}) & -\alpha(\Sigma)_{i,j}/h^2 & 0 \\ (\sigma_{21}(u^{[v]})_{i,j}) & (\sigma_{22}(u^{[v]})_{i,j}) & 0 & -\alpha(\Sigma)_{i,j}/h^2 \end{pmatrix}
\] (56)
\[
(\Sigma)_{i,j} = \begin{cases} \{ 2 @ (1, 1), (n, 1), (n, 1), \text{ and } (n, n) \text{ (four corners)} \} \\ \{ 3 \text{ for all } (1, j), (n, j), (i, 1), \text{ and } (i, n) \text{ where } 2 \leq i, j \leq n - 1 \text{ (boundary lines)} \} \\ \{ 4 \text{ for all } (i, j) \text{ where } 2 \leq i, j \leq n - 2 \text{ (interior points)} \} \end{cases}
\] (57)
\[
(G^\psi [z^{[v]}])_{i,j}^{[k+1/2]} = \begin{pmatrix} (g_1)_{i,j} + (1/h^2)(\Sigma)_{i,j}(u_1^{[v]+1})_{i,j}^{[k+1/2]} \\ (g_2)_{i,j} + (1/h^2)(\Sigma)_{i,j}(u_2^{[v]+1})_{i,j}^{[k+1/2]} \\ (g_3)_{i,j} - f_1(u_1^{[v]}, u_2^{[v]})_{i,j} + (\sigma_{11}(u^{[v]})_{i,j})u_1^{[v]}_{i,j} + (\sigma_{12}(u^{[v]})_{i,j})u_2^{[v]}_{i,j} + (\alpha/h^2)(\Sigma)_{i,j}(v_1^{[v]+1})_{i,j}^{[k+1/2]} \\ (g_4)_{i,j} - f_2(u_1^{[v]}, u_2^{[v]})_{i,j} + (\sigma_{21}(u^{[v]})_{i,j})u_1^{[v]}_{i,j} + (\sigma_{22}(u^{[v]})_{i,j})u_2^{[v]}_{i,j} + (\alpha/h^2)(\Sigma)_{i,j}(v_2^{[v]+1})_{i,j}^{[k+1/2]} \end{pmatrix},
\] (58)
and
\[
(\Sigma)_{i,j}(z^{[v]+1})_{i,j}^{[k+1/2]} = (z^{[v]+1})_{i,j}^{[k]} + (z^{[v]+1})_{i-1,j}^{[k]} + (z^{[v]+1})_{j+1,i}^{[k]} + (z^{[v]+1})_{i-1,j-1}^{[k]}.
\] (59)
With the above smoother, the curvature model can be solved using our Algorithms 7–7. Similarly to Section 3.2, we can show by the LFA that (1) \( \lim_{h \rightarrow 0} E_h(N^\psi \hat{e}(\hat{e})) = 1/256 \), i.e., the linearized system (52) is \( h \)-elliptic; and (2) \( \alpha \approx 1 \) (typically \( \alpha = 0.9725 \)) provides the good smoothing properties \( \mu_{boc}^* \approx 0.5 \), where the effects of \( \omega \) on smoothing factors of (55) in processing the registration problem shown in Fig. 3(a)–(b) are almost identical with Fig. 2. We also conducted several numerical tests to confirm that (52) is a robust smoother for the FAS-NMG method to solve the curvature model (47); see Section 5.2.

5. Numerical experiments

The main aim of this section is to show that our new Algorithm 7 using (25) is effective and robust in leading to convergent multigrid methods in the FAS-NMG framework for both the diffusion and curvature models. In all experiments, the bilinear interpolation was used to compute the transformed template image \( T(u) \).

5.1. The diffusion model tests

We first focus on the performance of Algorithm 7 for two sets of medical data, Examples 1 and 2 shown respectively in Fig. 3(a)–(b) and (d)–(e). We first consider its convergence behavior with different resolutions, and then show comparisons with several other methods. In all experiments for this section \( v_1 = 5, v_2 = 5, GSiter = 5 \), and \( \alpha = 0.9725 \) were used in this test.
Fig. 3. Registration results for X-ray and MRI images using the RDR method with Algorithms 2, 5, and 7. Left column: reference R, center column: template T, right column: the deformed template image T(u) obtained from Algorithm 7.

Table 1

<table>
<thead>
<tr>
<th>Algorithm 2</th>
<th>Algorithm 5</th>
<th>Algorithm 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>M/R/D/C</td>
<td>M/R/D/I/C</td>
<td>M/R/D/I/C</td>
</tr>
<tr>
<td>Example 1:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>α = 0.1000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>h = 1/128</td>
<td>10/2.1 × 10⁻⁵/0.1082/4.3</td>
<td>6/1.8 × 10⁻⁵/0.1082/7.9/10.1</td>
</tr>
<tr>
<td>h = 1/256</td>
<td>11/3.6 × 10⁻⁵/0.1161/22.5</td>
<td>6/3.1 × 10⁻⁵/0.1161/41.2/54.2</td>
</tr>
<tr>
<td>h = 1/512</td>
<td>11/2.8 × 10⁻⁵/0.1221/106.4</td>
<td>6/2.5 × 10⁻⁵/0.1221/180.1/231.9</td>
</tr>
<tr>
<td>h = 1/1024</td>
<td>11/8.6 × 10⁻⁵/0.1239/472.6</td>
<td>6/3.8 × 10⁻⁵/0.1239/798.1/1049.3</td>
</tr>
<tr>
<td>Example 2:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>α = 0.1176</td>
<td></td>
<td></td>
</tr>
<tr>
<td>h = 1/128</td>
<td>10/8.3 × 10⁻⁵/0.0522/4.2</td>
<td>5/1.5 × 10⁻⁵/0.0522/7.2/10.2</td>
</tr>
<tr>
<td>h = 1/256</td>
<td>11/5.6 × 10⁻⁵/0.0582/22.9</td>
<td>6/2.6 × 10⁻⁵/0.0582/31.3/41.7</td>
</tr>
<tr>
<td>h = 1/512</td>
<td>11/7.4 × 10⁻⁵/0.0615/108.3</td>
<td>7/5.4 × 10⁻⁵/0.0615/185.7/234.9</td>
</tr>
<tr>
<td>h = 1/1024</td>
<td>11/3.1 × 10⁻⁵/0.0633/478.1</td>
<td>7/6.9 × 10⁻⁵/0.0633/550.1/943.1</td>
</tr>
</tbody>
</table>

5.1.1. h-independent convergence tests

One of the key properties of multigrid techniques is that their convergence does not depend on the number of grid points. Thus, in the first test we designed our experiments to investigate this property with Algorithms 2, 5, and 7, and to back up our theoretical results by LFA in Section 3.2. The number of multigrid steps (V-cycles) used to drop the mean of the relative residual below ε₂ = 10⁻⁸, the relative reduction of dissimilarity, and runtimes (in seconds) are given in Table 1 with different sizes of grid points. The results show that all registration algorithms not only converge within a few multigrid steps, but they are also accurate because the dissimilarities between the reference and registered images have been reduced more than 87% for Example 1 and 94% for Example 2. For overall performance the experimental results suggest that Algorithm 7 would be preferred for practical applications because the multi-resolution idea used in the cooling process for α has been prove to be very useful for initialization.

5.1.2. Comparison with other multigrid methods

Methods in [4,12,18,21,22] are some existing unilevel or multigrid techniques used to solve the diffusion model.
In this section, we took Examples 1 to illustrate a comparison among our FAS-NMG method with Algorithm 2 and other six multigrid methods by starting with the fixed parameters $h = 1/256$, $\alpha = 1/8$, and $\mathbf{u}^{(0)} = 0$. Here we used $\tau = 10^{-2}$ and applied the so-called multi-resolution technique with the gradient descent methods of [4,21] for fairness. The FMG or LMG methods in [12,18,21] were performed using two pre-smoothing and two post-smoothing steps with the pointwise GS smoothers until the mean of residuals below a user-supplied threshold ($\text{tol}_{\text{LMG}} = 10^{-4}$).

Table 2 summarizes the results for all multigrid methods. As expected from the experiments, all methods are very fast and accurate in registering the given images because the dissimilarities between the reference and registered images have been reduced more than 80% within the first 20 iterations. However our method is not only the fastest way in solving the problem, but also in dropping the mean of the relative residuals below $10^{-8}$.

### 5.1.3. Comparison with the AOS method [4]

The AOS method is one of the most widely used gradient descent techniques for diffusion image registration [4,19,21]. We take Example 1 to illustrate a comparison with our FAS-NMG method. Table 3 summarizes the results for Algorithm 2 and AOS methods with different numbers of grid points. We used Algorithm 2 with $\alpha = 1/8$ for fairness (as Algorithm 7 will be even better). That is, we started both methods with the same $\alpha$ and the same initial guess, $\mathbf{u}^{(0)} = 0$. For the AOS method, the time-step $\tau$ is required to be sufficiently small for each size of the problem. We used $\tau = 10^{-2}$ for $h = 1/128 - 1/1024$. As expected from the experiments, both methods are very accurate in registering the given images because the dissimilarities between the reference and registered images have been reduced more than 85%. However the AOS fails to drop the relative residual to $10^{-8}$ in a few time steps (even for $\tau > 10^{-2}$) and the runtimes used by the RDR method is much faster than those of the AOS technique in delivering the same level of the relative dissimilarity.

### 5.1.4. Comparison of multigrid methods with different smoothers

Our aim in this section is to show that we have proposed a robust smoother for the FAS-NMG technique in solving the discrete system represented in (14). To end this, we have conducted several experiments of our FAS multigrid with different kinds of smoothers:

(i) the proposed smoother based on the RFP method [26] represented by Algorithm 3 in Section 3

(ii) the GS smoother based on the SFP method given by (19) in Section 2.4.3 (the standard FP method defined in [3, p. 79] with the standard linear (inner) solver)

(iii) the GS smoother based on the MSFP-FS method given by (20) and (21) in Section 2.4.3 (a modified SFP method adapted from the numerical techniques of Frohn-Schaufl et al. [8,9] with the standard linear solver)

(iv) the GS smoother based on MSFP-1 method given by (22) in Section 2.4.3 (a modified SFP method of the first type with the standard linear solver)

(v) the GS smoother based on MSFP-2 method given by (23) in Section 2.4.3 (a modified SFP method of the second type with the standard linear solver)
5.2. The curvature model

In this section, we aim to show that the FAS-NMG method with the smoother (52) based on our linearization idea (25) is effective and robust to solve the curvature model (47) within the multigrid framework similar to Algorithm 1.

To this end, we took only one data set of medical images, Examples 1, as shown in Fig. 3(a)–(b). We first investigate the convergence behavior and the registration accuracy of our FAS-NMG method with different sizes of image resolutions. Second we compare three numerical solution methods for solving the curvature model (47). In all experiments, $\Omega = [0, 1]^2$, $V = [0, 1]$, $v_1 = 10$, $v_2 = 10$, PCG$\textit{Siter} = 10$, and $\omega = 0.9725$ were used.

5.2.1. h-independent convergence tests

In this test, we started the registration processes with $\alpha = 10^{-4}$, $u^{(0)} = 0$, and $h = 1/128, \ldots, h = 1/1024$. The registered image is shown in Fig. 4(c). As expected from our LFA results, Fig. 5(a)–(b) shows that our FAS-NMG approach is h-independent. It takes only a few MG steps (almost the same number) to drop the mean of the relative residuals below $10^{-8}$. Moreover, as shown in Fig. 5(b) only one MG step can reduce the dissimilarities between the reference and the registered image more than 90% for the given registration problem.

5.2.2. Comparison with two other methods

In this section we aim to investigate the performance of the FAS-NMG method with the smoother (52) and those of other two methods: the FP method (52) (the smoother used as a stand-alone solver) and the DCT-based method in [6]. We note that the FT-based method in [33,34] is faster than the DCT-based method with the ratio of 2.3 for solving each linear time-dependent problem resulting from (47). Thus, for this class of gradient descent techniques it is enough to use only the DCT-based method, which is one of the most widely used techniques for the curvature model.
To this end, we started all methods with $h = 1/256$, $\alpha = 10^{-4}$ and $u^{(0)} = 0$. For the DCT-based method, the time-step $\tau$ was selected to be $\tau = 10^{-2}$ (since the fixed parameters $1/h^4 = N^4 = 256^4, V = [0, 1]$ and $\alpha = 10^{-4}$ were used in the discrete system of (47), $\tau = 10^{-2}$ was a reasonable time-step).

Compared with the results by other two methods, Fig. 6(b) shows that our FAS-NMG method is very fast in reducing the dissimilarities between the reference and registered images. Moreover, as shown in Fig. 6(a) it takes only a few steps to drop the mean of the relative residuals (MRR) below $10^{-8}$. This is a remarkable result to conclude that its performance in solving the curvature model (47) is much more efficient than those of the other two methods.

From both tests in Sections 5.1 and 5.2 they confirm that the smoothers based on our linearization idea by (25) are effective and robust in leading to convergent multigrid methods for both the diffusion and curvature models.

6. Conclusions

In this paper we first reviewed various iterative methods for solving image registration models and then addressed the problem of designing an optimal and efficient FAS-NMG technique. For the commonly used SSD model, we proposed a new, simple and yet effective fixed point smoothing scheme which is demonstrated to be more effective than a large class of other iterative methods for a diffusion model and a curvature model. A local Fourier analysis was done to confirm the effectiveness of the smoother. For an automatic way of selecting the regularization parameter $\alpha$, we proposed a multi-resolution idea that seems to work well for a range of problems. Numerical results confirmed the expected robustness and excellent speed in
References