

CLAIRE: SCALABLE GPU-ACCELERATED ALGORITHMS FOR DFFEOMORPHIC IMAGE REGISTRATION IN 3D

ANDREAS MANG

ABSTRACT. We present our work on scalable, GPU-accelerated algorithms for diffeomorphic image registration. The associated software package is termed CLAIRE. Image registration is a non-linear inverse problem. It is about computing a spatial mapping from one image of the same object or scene to another. In diffeomorphic image registration, the set of admissible spatial transformations is restricted to maps that are smooth, one-to-one, and have a smooth inverse. We formulate diffeomorphic image registration as a variational problem governed by transport equations. We use an inexact, globalized (Gauss-)Newton-Krylov method for numerical optimization. We consider semi-Lagrangian methods for numerical time integration. Our solver features mixed-precision, hardware-accelerated computational kernels for optimal computational throughput. We use the message-passing interface for distributed-memory parallelism and deploy our code on modern high-performance computing architectures. Our solver allows us to solve clinically relevant problems in under four seconds on a single GPU. It can also be applied to large-scale 3D imaging applications with data that is discretized on meshes with billions of voxels. We demonstrate that our numerical framework yields high-fidelity results in only a few seconds, even if we search for an optimal regularization parameter.

1. INTRODUCTION

In the present work, we discuss scalable, hardware-accelerated algorithms for diffeomorphic image registration. We review our past contributions and showcase results for a software framework termed CLAIRE [1–3]. Image registration is an ill-posed inverse problem [4]. It is a key methodology in medical image analysis. The inputs are two (or more, noisy) images $m_i \in \mathfrak{I}$, $i = 0, 1$, $\mathfrak{I} \subset \{u : \Omega \rightarrow \mathbb{R}\}$, of the same object or scene, compactly supported on some domain $\Omega \subseteq \mathbb{R}^d$, where $d \in \{2, 3\}$. In image registration, we seek a *plausible* spatial transformation $\mathbf{y} \in \mathfrak{Y}_{\text{ad}}$, $\mathfrak{Y}_{\text{ad}} \subset \{\phi : \mathbb{R}^d \rightarrow \mathbb{R}^d\}$, that maps points in the so-called *template* or *source image* m_0 to its corresponding points in the so-called *reference* or *target image* m_1 [4–6]. The notion of the plausibility of the map \mathbf{y} depends on the particular application. In the present work, we restrict the set \mathfrak{Y}_{ad} of admissible maps \mathbf{y} to \mathbb{R}^d -diffeomorphisms [7]. That is, $\mathfrak{Y}_{\text{ad}} \subseteq \text{diff}(\mathbb{R}^d)$, where $\text{diff}(\mathbb{R}^d)$ is the set of \mathbb{R}^d -diffeomorphisms, i.e., smooth maps from \mathbb{R}^d to \mathbb{R}^d that are one-to-one and onto, with a smooth inverse. The set $\text{diff}(\mathbb{R}^d)$ is closed under composition and taking the inverse; it forms a group. In this framework, deforming the template image m_0 corresponds to a change of coordinates $m_0 \circ \mathbf{y}^{-1}$; the image intensity in the transformed image $m_0 \circ \mathbf{y}^{-1}$ at coordinate $\mathbf{y}(\mathbf{x}) \in \mathbb{R}^d$ is identical to the value at the location $\mathbf{x} \in \mathbb{R}^d$ in the original image. Using this notation, the diffeomorphic image registration problem can be formulated as the problem of finding $\mathbf{y} \in \text{diff}(\mathbb{R}^d)$ such that $m_0 \circ \mathbf{y}^{-1} = m_1$. We illustrate this in Figure 1. We summarize the main notation and acronyms in Table 1.

1.1. Outline of the Method. The approach considered in the present work is related to a mathematical framework referred to as LDDMM [7–11]. We consider PDE-constrained optimization problems [12–15] governed by transport equations for diffeomorphic image registration. The transport map is parameterized by a smooth space-time field $\mathbf{v} \in \mathfrak{V}$, $\mathfrak{V} := L^q([0, 1], \mathfrak{H})$, $q \in \mathbb{N}$, where \mathfrak{H} is a Sobolev space of suitable regularity, i.e., $\mathfrak{H} = W^{p,s}(\Omega, \mathbb{R}^d)$, $p \in \mathbb{N}$, $s \in \mathbb{N}$. Our problem

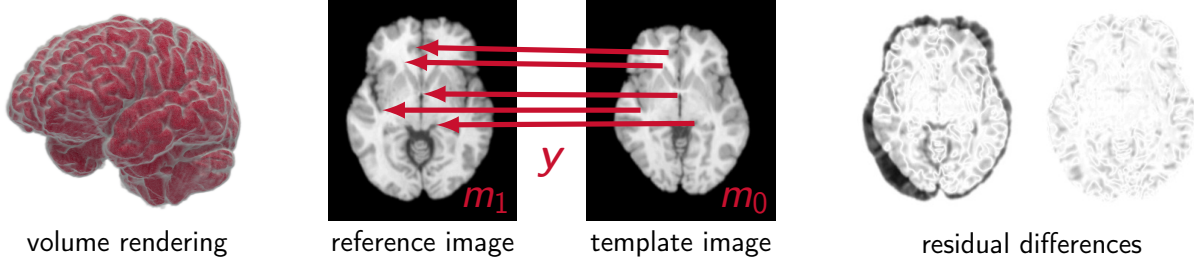


FIGURE 1. Image registration problem. On the left, we show a volume rendering of a 3D brain MRI. The figures in the middle show an axial slice of two MRI brain scans of different individuals. In image registration, we seek a map $\mathbf{y} \in \mathfrak{Y}_{\text{ad}} \subset \{\phi : \mathbb{R}^d \rightarrow \mathbb{R}^d\}$, $d \in \{2, 3\}$, that establishes a *plausible* spatial correspondence between these two images. In this work, we restrict the set of admissible spatial transformations \mathfrak{Y}_{ad} to \mathbb{R}^d -diffeomorphisms. On the right, we show the residual differences between the axial slices of the images shown in the middle before (left) and after (right) diffeomorphic (deformable) registration. Here, white represents small residual differences, and black indicates large residual differences. We note that the registration of two brains from different individuals is a common application for diffeomorphic image registration in computational anatomy. However, strictly speaking this example is in violation with our underlying assumptions; we do not register the “same object”—we register images of brains of different individuals (in an attempt to study anatomical variability).

TABLE 1. Notation, symbols, and acronyms.

Symbol/Acronym	Meaning
$d \in \mathbb{N}$	dimensionality of the ambient space
$\Omega \subset \mathbb{R}^d$	spatial domain
$m_0 : \bar{\Omega} \rightarrow \mathbb{R}$	template image
$m_1 : \bar{\Omega} \rightarrow \mathbb{R}$	reference image
$m : [0, 1] \times \bar{\Omega} \rightarrow \mathbb{R}$	state variable (transported image intensities)
$\mathbf{v} : \bar{\Omega} \rightarrow \mathbb{R}^d$	control variable (stationary velocity field)
$\lambda : [0, 1] \times \bar{\Omega} \rightarrow \mathbb{R}$	dual variable
$\text{dist} : \mathfrak{I} \times \mathfrak{I} \rightarrow \mathbb{R}$	distance functional
$\text{reg} : \mathfrak{V} \rightarrow \mathbb{R}$	regularization functional
$\mathcal{L} : \mathfrak{V} \rightarrow \mathfrak{V}^*$	regularization operator
$\alpha \in \mathbb{R}$	regularization parameter
\mathfrak{I}	orbit
\mathfrak{G}	group of diffeomorphisms
$\text{diff}(\mathbb{R}^d)$	set of \mathbb{R}^d -diffeomorphisms
FFT	Fast Fourier Transform
GMRES	Generalized Minimal RESidual (method)
GPU	Graphics Processing Unit
HPC	High Performance Computing
KKT	Karush–Kuhn–Tucker (conditions)
LDDMM	Large Deformation Diffeomorphic Metric Mapping
MRI	Magnetic Resonance Imaging
MPI	Message Passing Interface
ODE	Ordinary Differential Equation
PCG	Preconditioned Conjugate Gradient (method)
PDE	partial differential equation
RK2	second-order Runge–Kutta (method)
SL	Semi-Lagrangian (method)

formulation is of the form

$$\begin{aligned} (1a) \quad & \underset{m \in \mathfrak{M}_{\text{ad}}, \mathbf{v} \in \mathfrak{V}_{\text{ad}}}{\text{minimize}} && \text{dist}(m(1), m_1) + \text{reg}(\mathbf{v}) \\ (1b) \quad & \text{subject to} && c(m, \mathbf{v}) = 0. \end{aligned}$$

Here, $c : \mathfrak{M} \times \mathfrak{V} \rightarrow \mathfrak{Q}$ represents a PDE constraint. It is of the general form $c(m, \mathbf{v}) = \mathcal{A}(m, \mathbf{v}) - q$. The *parameter-to-observation map* $f : \mathfrak{V} \rightarrow \mathfrak{I}$ (i.e., the solution operator for the constraint) is formally given by $f(\mathbf{v}) = \mathcal{Q}\mathcal{A}^{-1}(m, \mathbf{v})q$. Here, \mathcal{Q} denotes the observation operator, i.e., a mapping that takes the output of \mathcal{A}^{-1} and maps it to “locations” at which data is available. The functional $\text{dist} : \mathfrak{I} \times \mathfrak{I} \rightarrow \mathbb{R}$ measures the discrepancy between the deformed template image $m(1) = f(\mathbf{v})$ and the reference image m_1 . The functional $\text{reg} : \mathfrak{V} \rightarrow \mathbb{R}$ is a regularization functional. We specify the precise choices in greater detail below.

We use the method of Lagrange multipliers to solve (1). We consider an *optimize-then-discretize* approach. We use a globalized, inexact reduced space (Gauss-)Newton–Krylov method for numerical optimization. We solve the PDEs that appear in the optimality conditions based on a SL method. The main computational kernels of our algorithm are interpolation and numerical differentiation. For interpolation, we use a Lagrange polynomial. For numerical differentiation, we consider a mixture of high-order finite difference operators and a pseudo-spectral method. We use MPI for distributed-memory parallelism and deploy our code on dedicated GPU architectures.

1.2. Related Work. We consider a PDE-constrained optimization problem for velocity-based diffeomorphic image registration. We refer to [12–14, 16–18] for insights into theory and algorithmic developments related to PDE-constrained optimization. Additional information about image registration and related work can be found in [4–7, 19, 20]. As we mentioned above, we restrict ourselves to diffeomorphic image registration. An intuitive approach to safeguard against non-diffeomorphic maps \mathbf{y} is to add hard and/or soft constraints to the variational problem [21–24]. An alternative strategy is to introduce a pseudo-time variable t and invert for a smooth velocity field \mathbf{v} that parameterizes \mathbf{y} [7, 8, 10, 11, 25–29]; our approach falls into this category. In [8–11, 26, 29], the flow of the sought after diffeomorphism \mathbf{y} is modelled as the solution of the ODE $\partial_t \phi = \mathbf{v} \circ \phi$ for $t \in (0, 1]$ with initial condition $\phi = \text{id}_{\mathbb{R}^d}$ at time $t = 0$, where \mathbf{v} is a smooth, time-dependent vector field from \mathbb{R}^d to \mathbb{R}^d and $\text{id}_{\mathbb{R}^d} : \mathbb{R}^d \rightarrow \mathbb{R}^d$, $\text{id}_{\mathbb{R}^d}(\mathbf{x}) = \mathbf{x}$, is the identity transformation in \mathbb{R}^d . This ODE enters the variational problem as a constraint; we arrive at a non-linear optimal control problem with state variable ϕ and control \mathbf{v} . The sought-after diffeomorphism \mathbf{y} that maps one image to another corresponds to the end point of the flow ϕ , i.e., $\mathbf{y} = \phi(t = 1)$. This approach is commonly referred to as LDDMM [8]. We describe it in greater detail in the main part of this manuscript. In our formulation, the diffeomorphism \mathbf{y} does no longer appear; we do *not* model the deformed template image as the application of \mathbf{y} to m_0 . Instead, we transport the intensities of the template image m_0 given some candidate \mathbf{v} based on a hyperbolic transport equation [30–32]. Unlike most existing approaches, our framework features explicit control on volume change introduced by the mapping by controlling the divergence of \mathbf{v} . This formulation was originally proposed in [33]; a similar approach is described in [32]. Works of other groups that consider divergence-free velocities \mathbf{v} in similar contexts have been described in [34–38].

Our formulation has been introduced in [30, 33]. The work most closely related to ours in terms of the problem formulation is [31, 32, 39–44]. Related formulations for optimal mass transport are discussed in [44–48]. In contrast to optimal mass transport, our formulation keeps the transported quantities constant along the characteristics, i.e., mass is *not* preserved. Our formulation is related to traditional optical flow formulations [37, 49, 50]. The main difference is that the transport equation for the image intensities of m_0 enters our formulation as a hard constraint. PDE-constrained formulations for optical flow that are equivalent to our formulation are described in [32, 34, 51, 52].

Among the most popular packages for diffeomorphic registration are **Demons** [27, 53], **ANTs** [54, 55], **Deformetrica** [56, 57], or **DARTEL** [58]. There are only few works on effective numerical methods

for velocity-based diffeomorphic image registration, and even fewer on scalable algorithms. Works of other groups on numerical algorithms for the solution of diffeomorphic association problems (for images as well as surface representations) are, e.g., described in [8, 31, 43, 59–64]. The majority of existing works consider an *optimize-then-discretize* approach for solving the variational problem [8, 31, 43, 64–67]; *discretize-then-optimize* approaches for related problem formulations can be found in [48, 62, 68–70]. In the work discussed in this exposition, we also consider an *optimize-then-discretize* approach [30, 33, 71]; an implementation for a *discretize-then-optimize* approach for problem formulations similar to the one considered here can be found in [48].

Despite the fact that first-order methods for optimization have poor convergence rates for non-linear, ill-posed inverse problems, most work on algorithms for formulations similar to ours, with the exception of ours [15, 30, 33, 48, 71, 72] and [27, 44, 45, 61, 66, 73, 74], use first order gradient descent-type approaches. Work on operator-splitting algorithms for LDDMM (and related problems) can be found in [69, 70, 75, 76]. Other recent works that do not explicitly derive optimality conditions based on variational principles but rely on automatic differentiation can be found in [56, 77–80]. Lastly, we note that the success of machine learning in various scientific disciplines has led to several recent works that attempt to solve the inverse problem of diffeomorphic registration based on machine learning techniques [80–89]. As we will show, our dedicated hardware-accelerated implementation [90, 91] allows us to solve diffeomorphic image registration problems in 3 to 4 seconds on a single GPU without considering machine learning approaches.

We consider a globalized, reduced space (Gauss-)Newton-Krylov method [1, 30]. For these methods to be effective, it is crucial to design a good preconditioner for solving the reduced space KKT system [92]. Related work on designing preconditioners for problems similar to ours can be found in [44, 45, 74]. Another key ingredient is fast algorithms to solve the PDEs that appear in the optimality systems. In our case, the most expensive PDE operators are (hyperbolic) transport equations. We refer to [8, 31, 32, 34, 45, 48, 61, 74] for different numerical methods to solve these types of PDEs in the context of PDE-constrained optimization. We use a SL method [8, 34, 71, 72].

What separates CLAIRE [1, 2] from most existing packages for velocity-based diffeomorphic image registration, aside from the numerics, is that it features hardware-accelerated computational kernels and that it has been deployed to dedicated HPC architectures [1, 71, 90, 91, 93]. Examples for parallel algorithms for PDE-constrained optimization problems can be found in [16, 94–100]. Surveys for parallel implementations of image registration algorithms are [101–104]. Many of these works consider low-dimensional parameterizations based on an expansion of the deformation map \mathbf{y} in terms of smooth basis functions. Examples of GPU implementations of these approaches are [105–107]. GPU implementations of formulations similar to ours are described in [47, 56, 108–111]. Our memory-distributed implementation uses MPI for parallelism and allows us to solve problems of unprecedented scale [33, 90, 93, 112]. The linear solvers and the optimizer are built on top of PETSc/TAO [113–116]. Our CPU implementation [1, 33, 93] allows us to solve problems with 3,221,225,472 unknowns in 2 min on 22 compute nodes (256 MPI tasks), and in less than 5 s if we use 342 compute nodes (4096 MPI tasks). Our GPU implementation [90, 91] allows us to solve clinically relevant problems (50,000,000 unknowns) in less than 5 s on a single GPU. Our multi-GPU implementation for large scale problems is described in [90] and applied to large scale (biomedical) imaging data in [112]. We limit the numerical results reported in this study to our GPU implementation [90, 91].

1.3. Contributions. Our contributions are as follows:

- We overview our work on CLAIRE—a memory distributed algorithmic framework for diffeomorphic image registration based on variational optimization problems governed by transport equations. In particular, we recapitulate our contributions presented in [1, 3, 15, 30, 33, 71, 72, 90, 91, 93].

- We report new results and study the performance of CLAIRE for real world medical imaging data in 3D [1–3]. We include results for different similarity measures—normalized cross correlation and the default squared L^2 -distance. In addition, we present results for an improved implementation in which we store the state variable and its gradient, reducing the runtime from roughly five seconds reported in prior work to slightly more than three seconds for clinically relevant problems (50,000,000 unknowns).

1.4. Limitations. CLAIRE has several limitations. First, CLAIRE only supports stationary velocity fields. Stationary velocities yield similar residuals in practical applications [30]. However, they are less expressive [48]; they only allow us to model a subset of the diffeomorphisms that can be modeled by using non-stationary \mathbf{v} . Second, CLAIRE only supports the registration of images acquired from the same modality. Implementing distance measures that allow for more complicated intensity relationships between images to be registered requires more work. Third, while our schemes for preconditioning the reduced space Hessian are effective and, in general, mesh-independent, they are not independent of the choice of the regularization parameter. Fourth, the GPU implementation only supports single precision. We trade numerical accuracy for computational throughput. For example, applying (the inverse of) high-order differential operators (e.g., biharmonic operators) results in significant numerical round-off errors and, consequently, is currently not supported. This, in conjunction with other algorithmic choices does not allow us to solve the optimization problem to arbitrary accuracy, in particular for practical parameter choices. Nonetheless, we can see that in practice we obtain an excellent agreement between the registered datasets even if the gradient of our problem is not driven to zero.

1.5. Outline. We present the formulation and numerical methods in §2. This includes a discussion of the mathematical framework that motivates our approach (see §2.1), a brief recapitulation of the general problem formulation (see §2.2), the optimality conditions (see §2.3), and the Newton step (see §2.4), followed by a presentation of our numerical approach (see §2.5). We present some numerical results in §3 and conclude with §4.

2. METHODS

In the following, we present the problem formulation as well as our numerical approach and implementation aspects. We start with discussing some background material related to the considered problem formulation.

2.1. Mathematical Foundations. Our problem formulation is related to LDDMM [7–10, 117, 118]—a mathematical framework for diffeomorphic image registration and shape matching. It builds upon the seminal work [119–121].

Let $\mathbf{k} = (k_1, \dots, k_d) \in \mathbb{N}^d$, $d \in \{1, 2, 3\}$, denote a multi-index, and let

$$\partial^{\mathbf{k}} := \frac{\partial^{|\mathbf{k}|}}{\partial_1^{k_1} \dots \partial_d^{k_d}}$$

denote the differential operator of order $|\mathbf{k}| = \sum_{i=1}^d k_i$. Here, ∂_i denotes the partial derivative with respect to the coordinate direction x_i with $\mathbf{x} := (x_1, \dots, x_d) \in \Omega$ defined on some domain $\Omega \subseteq \mathbb{R}^d$. Moreover, let $q \in \mathbb{N}$, $1 \leq p \leq \infty$. We denote by

$$C^q(\Omega) := \left\{ u : \Omega \rightarrow \mathbb{R} : \partial^{\mathbf{k}} u \text{ is continuous for } |\mathbf{k}| \leq q \right\}$$

the space of q -times continuously differentiable functions on Ω . Moreover, let

$$W^{q,p}(\Omega) := \left\{ u \in L^p(\Omega) : \partial^{\mathbf{k}} u \in L^p(\Omega) \text{ for } 0 \leq |\mathbf{k}| \leq q \right\}$$

denote the Sobolev space with norm

$$(2) \quad \|u\|_{q,p} := \begin{cases} \left(\sum_{0 \leq |\mathbf{k}| \leq q} \|\partial^{\mathbf{k}} u\|_p^p \right)^{1/p} & \text{if } 1 \leq p < \infty, \\ \max_{0 \leq |\mathbf{k}| \leq q} \|\partial^{\mathbf{k}} u\|_\infty & \text{if } p = \infty. \end{cases}$$

Here, $\|\cdot\|_\infty$ denotes the standard supremum norm. Using these definitions, we denote by $C_0^q(\Omega)^d \subset C^q(\Omega)^d$ with $q \in \mathbb{N}$ the completion of the space of vector fields of class C^q which along with their derivatives of order less than or equal to q converge to zero at infinity. The space $C_0^q(\Omega)^d$ is a Banach space for the norm $\|u\|_{q,\infty}$. Similarly, we define the Sobolev space $W_0^{q,p}(\Omega)^d$ as a space that consists of elements with compact support on $\Omega \subseteq \mathbb{R}^d$.

We introduce a pseudo-time variable $t \in [0, 1]$, a suitable Hilbert space \mathfrak{H} of smooth vector fields in \mathbb{R}^d , and parameterize diffeomorphisms using smooth vector fields $\mathbf{v} \in \mathfrak{V}$, $\mathfrak{V} := L^r([0, 1], \mathfrak{H})$, $1 \leq r \leq \infty$, $t \mapsto \mathbf{v}_t := \mathbf{v}(t, \cdot)$, $\mathbf{v}_t \in \mathfrak{H}$. This allows us to model the flow of \mathbb{R}^d -diffeomorphisms $\phi_t := \phi(t, \cdot)$ as the solution of the ODE

$$(3) \quad \begin{aligned} \partial_t \phi_t &= \mathbf{v}_t \circ \phi_t & \text{for } t \in (0, 1], \\ \phi_t &= \text{id}_{\mathbb{R}^d} & \text{for } t = 0, \end{aligned}$$

where $\text{id}_{\mathbb{R}^d} : \mathbb{R}^d \rightarrow \mathbb{R}^d$, $\text{id}_{\mathbb{R}^d}(\mathbf{x}) = \mathbf{x}$, is the identity transformation in \mathbb{R}^d and the vector field \mathbf{v}_t tends to zero as $\mathbf{x} \rightarrow \infty$; that is, we assume $\mathbf{v}_t \in C_0^q(\Omega)^d \supseteq \mathfrak{H}$ for any $t \in [0, 1]$. This assumption, along with suitable regularity requirements in time, guarantees that (3) admits a unique solution. Moreover, it is ensured that solutions of (3) are \mathbb{R}^d -diffeomorphisms [7].

We assume L^1 -integrability in time, i.e., $r = 1$ [7]. The differentiability class q and the integrability order p of the Sobolev norm (2) are chosen to stipulate adequate regularity requirements in space. A common choice for p is $p = 2$. The choice of q depends on the dimension d of the ambient space Ω . In general, we have $d \in \{1, 2, 3\}$. Based on the Sobolev embedding theorem [122] we observe that for $p = 2$ and $q > (d/2) + 1$ the embedding $W_0^{q,2}(\Omega) \hookrightarrow C^1(\bar{\Omega})$ is compact. Since this embedding holds for all components of \mathbf{v}_t , we have that $\mathbf{v}_t \in \mathfrak{H} = W^{q,2}(\Omega)^d$ with $q > 5/2$ for $d = 3$ is an admissible space that yields a diffeomorphic flow ϕ_t of smoothness class $1 \leq s < q - (3/2)$, $s \in \mathbb{N}$. We refer to [7, 11, 118] for a more rigorous discussion.

The set of all endpoints $\mathbf{y} := \phi_1$ at time $t = 1$ of admissible flows ϕ_t is a subgroup

$$\mathfrak{G} := \left\{ \phi_1 : \int_0^1 \|\mathbf{v}_t\|_{\mathfrak{H}} dt < \infty \right\}$$

of C^s -diffeomorphisms in \mathbb{R}^d . This subgroup can be equipped with a right-invariant metric defined as the minimal path length of all geodesics joining two elements in \mathfrak{G} [9, 25, 123, 124]. The geodesic distance between $\text{id}_{\mathbb{R}^d}$ and a mapping $\mathbf{y} \in \mathfrak{G}$ corresponds to the square root of the kinetic energy

$$(4) \quad \text{kin}(\mathbf{v}) := \|\mathbf{v}\|_{L^2([0,1], \mathfrak{H})}^2 = \int_0^1 \|\mathbf{v}_t\|_{\mathfrak{H}}^2 dt$$

subject to the constraint that \mathbf{y} is equal to the solution ϕ of (3) at time $t = 1$ for the energy minimizing velocity \mathbf{v} . We denote this geodesic distance by $\text{dist}_{\mathfrak{G}}(\text{id}_{\mathbb{R}^d}, \mathbf{y})$,

$$\text{dist}_{\mathfrak{G}}(\text{id}_{\mathbb{R}^d}, \mathbf{y})^2 := \inf_{\mathbf{v} \in \mathfrak{V}} \left\{ \int_0^1 \|\mathbf{v}_t\|_{\mathfrak{H}}^2 dt : \mathbf{v} \in \mathfrak{V}, \mathbf{y} = \phi_1, \partial_t \phi_t = \mathbf{v}_t \circ \phi_t, \phi_0 = \text{id}_{\mathbb{R}^d} \right\}.$$

The geodesic distance between two maps \mathbf{y} and ψ is given by $\text{dist}_{\mathfrak{G}}(\text{id}_{\mathbb{R}^d}, \mathbf{y} \circ \psi^{-1})$.

Similarly, we can measure the geodesic distance between two images $m_0, m_1 \in \mathfrak{I}$ in terms of the kinetic energy associated with the energy minimizing \mathbf{v} that gives rise to the diffeomorphic flow ϕ that maps m_0 to m_1 . To do so, we assume that the image m_1 is in the orbit \mathfrak{I} of the template image m_0 for the group \mathfrak{G} of diffeomorphism, where

$$\mathfrak{I} := \{m : \Omega \rightarrow \mathbb{R} : m = m_0 \circ \mathbf{y}^{-1}, \mathbf{y} \in \mathfrak{G}\}.$$

Using the geodesic distance $\text{dist}_{\mathfrak{G}}$ introduced above we have

$$\text{dist}_{\mathfrak{J}}(m_0, m_1) := \inf_{\mathbf{y} \in \mathfrak{G}} \{ \text{dist}_{\mathfrak{G}}(\text{id}_{\mathbb{R}^d}, \mathbf{y}) : m_1 = m_0 \circ \mathbf{y}^{-1} \},$$

where \mathbf{y} corresponds to the endpoint of the flow ϕ . This notion of measuring distances between deformable objects has led to the emergence of a field of study in medical image analysis referred to as *computational anatomy* [123, 125–128].

Putting everything together, we can formulate the diffeomorphic matching of the template image m_0 to the reference image m_1 as a variational optimization problem. We stated initially that we seek a diffeomorphic map \mathbf{y} such that $m_0 \circ \mathbf{y}^{-1} = m_1$. This is an ill-posed problem; we try to estimate a vector field given scalar data. Consequently, a solution \mathbf{y} may not exist, and if it exists, it may not be unique or depend continuously on the data. To alleviate the ill-posedness we introduce a regularization model that rules out unwanted solutions. For example, we can restrict ourselves to maps \mathbf{y} that are close to the identity $\text{id}_{\mathbb{R}^d}$, i.e., we penalize the distance between $\text{id}_{\mathbb{R}^d}$ and \mathbf{y} . To alleviate existence issues, we relax the exact matching requirement to $m_0 \circ \mathbf{y}^{-1} \approx m_1$. We do so by introducing a distance that measures the proximity between the deformed template image $m_0 \circ \mathbf{y}^{-1}$ and the reference image m_1 . In conclusion, we seek $\mathbf{y} \in \mathfrak{G} \subset \text{diff}(\mathbb{R}^d)$ as a solution to

$$\underset{\mathbf{y} \in \mathfrak{G}}{\text{minimize}} \quad \text{dist}(m_0 \circ \mathbf{y}^{-1}, m_1) + \frac{\alpha}{2} \text{dist}_{\mathfrak{G}}(\mathbf{y}, \text{id}_{\mathbb{R}^d}).$$

We can reformulate the variational problem above as an optimal control problem governed by (3) [8, 11]. We have

$$(5a) \quad \underset{\phi \in \mathfrak{F}_{\text{ad}}, \mathbf{v} \in \mathfrak{V}_{\text{ad}}}{\text{minimize}} \quad \text{dist}(m_0 \circ \phi_1^{-1}, m_1) + \frac{\alpha}{2} \text{kin}(\mathbf{v})$$

$$(5b) \quad \begin{aligned} &\text{subject to } \partial_t \phi_t = \mathbf{v}_t \circ \phi_t \quad \text{in } (0, 1], \\ &\quad \phi_t = \text{id}_{\mathbb{R}^d} \quad \text{for } t = 0, \end{aligned}$$

where the first term in the objective functional measures the discrepancy between the deformed template image $m_0 \circ \phi_1^{-1}$ and the reference image m_1 , the second term denotes the kinetic energy in (4) and the parameter $\alpha > 0$ balances their contribution. The norm $\|\mathbf{v}_t\|_{\mathfrak{H}}^2$ in the definition of the kinetic energy in (4) is typically modelled as

$$\|\mathbf{v}\|_{\mathfrak{H}}^2 = \langle \mathbf{v}, \mathbf{v} \rangle_{\mathfrak{H}} = \langle \mathcal{B}\mathbf{v}, \mathcal{B}\mathbf{v} \rangle_{\mathbb{R}^d} = \langle \mathcal{L}\mathbf{v}, \mathbf{v} \rangle_{\mathbb{R}^d},$$

where $\mathcal{L} : \mathfrak{V} \rightarrow \mathfrak{V}^*$, $\mathcal{L} = \mathcal{B}^* \mathcal{B}$, is a differential operator of adequate order. A common choice for \mathcal{B} is a symmetric, positive definite Helmholtz operator of the form $\mathcal{B} := (\beta \text{id} - \Delta_d)^\gamma$, $\beta, \gamma > 0$ [8], where $\Delta_d \mathbf{u}(\mathbf{x}) := (\Delta u_1(\mathbf{x}), \dots, \Delta u_d(\mathbf{x}))$, $\Delta := \sum_{i=1}^d \partial_{ii}$ for any $\mathbf{u} : \bar{\Omega} \rightarrow \mathbb{R}^d$.

Other data structures than images that can be registered within this framework are landmarks [129, 130], curves [131, 132], surfaces [64, 68–70, 79, 131, 133, 134], tensor fields [135] or functional data on manifolds. We refer to [8, 62, 68–70] for numerical methods to solve the control problem in (5).

2.2. Variational Problem Formulation. In this section, we review the problem formulation considered in CLAIRE. We assume that $m_i : \Omega \rightarrow \mathbb{R}$, $i = 0, 1$, are smooth C^1 -functions compactly supported on Ω . As stated in §1, we formulate diffeomorphic image registration as a PDE-constrained optimization problem of the general form (1). This is different from the ODE-constrained optimization problem (5). Motivated by the formulation discussed above, we introduce a pseudo-time variable $t \in [0, 1]$ and invert for a smooth, time-dependent velocity field $\mathbf{v} \in \mathfrak{V}$ [30]. However, to reduce the computational complexity, our hardware-accelerated implementation no longer inverts for a time-dependent velocity \mathbf{v} but for a *stationary velocity field* $\mathbf{v} : \Omega \rightarrow \mathbb{R}^d$. This not only reduces the complexity of the optimization problem but also simplifies the implementation. We note that stationary velocities no longer define a Riemannian metric as described in §2.1. However, we still generate diffeomorphic transformations. Moreover, we did not observe a deterioration in

registration accuracy when comparing results to a non-stationary implementation [30]. Related work by other groups that use stationary \mathbf{v} can be found in [27, 58, 136–139].

In its simplest form, the PDE constraint c in (1) for stationary \mathbf{v} is given by the *hyperbolic transport equation*

$$(6a) \quad \partial_t m(t, \mathbf{x}) + \nabla m(t, \mathbf{x}) \cdot \mathbf{v}(\mathbf{x}) = 0 \quad \text{in } (0, 1] \times \Omega,$$

$$(6b) \quad m(t, \mathbf{x}) = m_0(\mathbf{x}) \quad \text{in } \{0\} \times \Omega.$$

The solution of this PDE is the transported intensities m of the template image m_0 . The endpoint $m(1) := m(1, \cdot)$ at time $t = 1$ corresponds to the deformed template image for some trial velocity \mathbf{v} .

The second building block of our variational problem formulation is the distance functional $\text{dist} : \mathfrak{I} \times \mathfrak{I} \rightarrow \mathbb{R}$ in (1a) that quantifies the discrepancy between the transported template image $m(1) := m(\cdot, 1)$ at time $t = 1$ and the reference image m_1 . A common choice for this terminal (endpoint) cost is given by the squared L^2 -distance

$$\text{dist}(m(1), m_1) = \frac{1}{2} \int_{\Omega} (m(1, \mathbf{x}) - m_1(\mathbf{x}))^2 d\mathbf{x}.$$

While this is a common choice in many diffeomorphic image registration packages [1, 8], this distance measure can only be used for registering images acquired using the same imaging modality. We present an alternative in the appendix.

The last building block is the regularization functional $\text{reg} : \mathfrak{V} \rightarrow \mathbb{R}$. Motivated by the problem formulation presented in §2.1, we use

$$\text{reg}(\mathbf{v}) = \frac{\alpha}{2} \|\mathbf{v}\|_{\mathfrak{H}}^2 = \frac{\alpha}{2} \langle \mathcal{L}\mathbf{v}, \mathbf{v} \rangle_{\mathbb{R}^d},$$

where $\mathcal{L} : \mathfrak{V} \rightarrow \mathfrak{V}^*$ is a differential operator of adequate order. CLAIRe, in general, features H^1 -, H^2 - and H^3 -norms and semi-norms for the regularization of \mathbf{v} [1, 30, 33]. The default regularization operator is an H^1 -seminorm, i.e., $\mathcal{L} = -\Delta_d$, with an additional H^1 -norm that penalizes the divergence of the velocity [1, 90, 91, 112]. We provide additional details in the appendix.

Putting everything together, we arrive at the PDE-constrained optimization problem

$$(7a) \quad \begin{aligned} & \underset{m \in \mathfrak{M}_{\text{ad}}, \mathbf{v} \in \mathfrak{V}_{\text{ad}}}{\text{minimize}} && \frac{1}{2} \int_{\Omega} (m(1, \mathbf{x}) - m_1(\mathbf{x}))^2 d\mathbf{x} + \frac{\alpha}{2} \|\mathbf{v}\|_{\mathfrak{H}}^2 \\ & \text{subject to} && \partial_t m(t, \mathbf{x}) + \nabla m(t, \mathbf{x}) \cdot \mathbf{v}(\mathbf{x}) = 0 \quad \text{in } (0, 1] \times \Omega, \\ & && m(t, \mathbf{x}) = m_0(\mathbf{x}) \quad \text{in } \{0\} \times \Omega. \end{aligned}$$

Similar problem formulations have been considered in [31, 32, 39, 40]. For simplicity, we discuss the numerical methods for the problem formulation in (7). However, we note that we considered different variants of this formulation in our past work [1, 30, 33, 90]. We discuss these in greater detail in the appendix.

2.3. Optimality Conditions. In the present work, we consider an *optimize-then-discretize* approach. The advantages of this approach are that the formal optimality conditions are straightforward to derive. They also retain interpretability; for example, we will see that the adjoint equation of the transport equation in (6) represents a continuity equation for the image mismatch (see (10)). Moreover, one can freely decide on the numerical methods to solve the PDEs associated with the optimality conditions. A disadvantage of this approach is that the discrete gradient is (in general) not consistent with the discretized objective functional (contingent on the numerical scheme used for discretization). Consequently, it is not possible to solve the variational optimization problem with arbitrary accuracy (i.e., to machine precision). A *discretize-then-optimize* approach guarantees that the discrete gradient is consistent with the discretized objective functional. However, depending on the discretization this approach also has drawbacks. We refer to [17] for a general

discussion and to [48, 62] for examples of *discretize-then-optimize* implementations for a problem of the form (7).

We consider the method of Lagrange multipliers to solve (1). We introduce the dual variable $\lambda : [0, 1] \times \bar{\Omega} \rightarrow \mathbb{R}$, $\lambda \in \mathfrak{M}^*$, for the transport equation (6). The Lagrangian functional is given by

$$(8) \quad \begin{aligned} \ell(\Xi) = & \frac{1}{2} \int_{\Omega} (m(1, \mathbf{x}) - m_1(\mathbf{x}))^2 d\mathbf{x} + \frac{\alpha}{2} \langle \mathcal{L}\mathbf{v}, \mathbf{v} \rangle_{\mathbb{R}^d} \\ & + \int_0^1 \langle \partial_t m + \nabla m \cdot \mathbf{v}, \lambda \rangle_{L^2(\Omega)} dt + \langle m(0) - m_0, \lambda \rangle_{L^2(\Omega)} \end{aligned}$$

where $\Xi := (\mathbf{v}, m, \lambda) \in \mathfrak{V} \times \mathfrak{M} \times \mathfrak{M}^*$.

Computing first variations with respect to the control variable \mathbf{v} yields the reduced gradient

$$(9) \quad \mathbf{g}[\mathbf{v}](\mathbf{x}) := \alpha \mathcal{L}[\mathbf{v}](\mathbf{x}) + \int_0^1 \lambda(t, \mathbf{x}) \nabla m(t, \mathbf{x}) dt.$$

To be able to evaluate the reduced gradient we require the state variable $m \in \mathfrak{M}$ and the dual variable $\lambda \in \mathfrak{M}^*$. We can find the state variable by solving (6) forward in time. Formally, this equation is obtained by computing the first variations of ℓ in (8) with respect to λ . The dual variable λ can be found by solving the adjoint equations backward in time. Formally, the adjoint equations are found by computing variations of ℓ in (8) with respect to m . We obtain

$$(10a) \quad -\partial_t \lambda(t, \mathbf{x}) - \nabla \cdot \lambda(t, \mathbf{x}) \mathbf{v}(\mathbf{x}) = 0 \quad \text{in } [0, 1] \times \Omega,$$

$$(10b) \quad \lambda(t, \mathbf{x}) = -(m(1, \mathbf{x}) - m_1(\mathbf{x})) \quad \text{in } \{1\} \times \Omega,$$

subject to periodic boundary conditions on $\partial\Omega$. Notice that this equation represents a continuity equation; we transport the mismatch between the deformed template image $m(1, \cdot)$ and the reference image m_1 backward in time. If we change the distance measure $\text{dist} : \mathfrak{I} \times \mathfrak{I} \rightarrow \mathbb{R}$ in (7), the final conditions in (10b) will change.

2.4. Newton Step. We consider a (Gauss–)Newton–Krylov method for numerical optimization [30]. We provide more details in §2.5. The PDE operators associated with the Hessian can be found by formally computing second-order variations of the Lagrangian ℓ in (8). The expression for the Hessian matvec—i.e., the application of the Hessian to a vector $\tilde{\mathbf{v}}$ —is given by

$$(11) \quad \begin{aligned} \mathcal{H}[\tilde{\mathbf{v}}](\mathbf{x}) &= \mathcal{H}_{\text{reg}}[\tilde{\mathbf{v}}](\mathbf{x}) + \mathcal{H}_{\text{dat}}[\tilde{\mathbf{v}}](\mathbf{x}) \\ &= \alpha \mathcal{L}[\tilde{\mathbf{v}}](\mathbf{x}) + \int_0^1 \left\{ \tilde{\lambda}(t, \mathbf{x}) \nabla m(t, \mathbf{x}) + \lambda(t, \mathbf{x}) \nabla \tilde{m}(t, \mathbf{x}) \right\} dt. \end{aligned}$$

The variable $\tilde{\mathbf{v}} : \bar{\Omega} \rightarrow \mathbb{R}^d$, $\mathbf{v} \in \mathfrak{V}$, represents the incremental control variable, i.e., the search direction for \mathbf{v} . The operators \mathcal{H}_{reg} and \mathcal{H}_{dat} denote the regularization part and the data part of the reduced space Hessian, respectively. For the latter, the dependence on $\tilde{\mathbf{v}}$ is hidden in the incremental PDE operators. Given a candidate \mathbf{v} and a candidate $\tilde{\mathbf{v}}$ we require the state variable $m : [0, 1] \times \bar{\Omega} \rightarrow \mathbb{R}$, the dual variable $\lambda : [0, 1] \times \bar{\Omega} \rightarrow \mathbb{R}$, the incremental state variable $\tilde{m} : [0, 1] \times \bar{\Omega} \rightarrow \mathbb{R}$, and the incremental adjoint variable $\tilde{\lambda} : [0, 1] \times \bar{\Omega} \rightarrow \mathbb{R}$. We can find the state and dual variables during the evaluation of the reduced gradient in (9). The incremental state variable can be found by solving

$$(12a) \quad \partial_t \tilde{m}(t, \mathbf{x}) + \nabla \tilde{m}(t, \mathbf{x}) \cdot \mathbf{v}(t, \mathbf{x}) + \nabla m(t, \mathbf{x}) \cdot \tilde{\mathbf{v}}(t, \mathbf{x}) = 0 \quad \text{in } (0, 1] \times \Omega,$$

$$(12b) \quad \tilde{m}(t, \mathbf{x}) = 0 \quad \text{in } \{0\} \times \Omega,$$

subject to periodic boundary conditions on $\partial\Omega$ forward in time. We can find the incremental dual variable $\tilde{\lambda}$ by solving

$$(13a) \quad -\partial_t \tilde{\lambda}(t, \mathbf{x}) + \nabla \cdot (\tilde{\lambda}(t, \mathbf{x}) \mathbf{v}(\mathbf{x}) + \lambda(t, \mathbf{x}) \tilde{\mathbf{v}}(\mathbf{x})) = 0 \quad \text{in } [0, 1] \times \Omega,$$

$$(13b) \quad \tilde{\lambda}(t, \mathbf{x}) = -\tilde{m}(t, \mathbf{x}) \quad \text{in } \{1\} \times \Omega,$$

subject to periodic boundary conditions on $\partial\Omega$ backward in time. Consequently, each time we apply \mathcal{H} to a vector, we have to solve two PDEs.

2.5. Numerics. The numerical implementation discussed below is based on the computational kernels described in [1, 30, 71, 72, 90, 91, 93]. The hardware-accelerated CPU implementation is described in [1, 71, 93]. The GPU implementation is described in [90, 91].

We note that the GPU implementation is only available in single precision. This poses several numerical challenges. In particular, we observed that our scheme does not allow us to solve the optimization problem to arbitrary accuracy. This is due to the accumulation of numerical errors, dominated by the time integration and numerical differentiation. Moreover, the numerical gradient is inconsistent with the objective function. This is caused by our particular choice of a numerical time integrator in conjunction with an optimize-then-discretize approach.

2.5.1. Discretization. We consider a nodal discretization in space. That is, we subdivide the spatial interval $\Omega = [-\pi, \pi]^d \subset \mathbb{R}^d$ into $n_i \in \mathbb{N}$ cells of width $h_i > 0$, $i = 1, \dots, d$, along each spatial direction x_i , $i = 1, \dots, d$. The width of the cells $\mathbf{h} = (h_1, \dots, h_d) \in \mathbb{R}^d$ along each spatial direction is given by $h_i = 2\pi/n_i$, $i = 1, \dots, d$. Let $\mathbf{x}_l \in \mathbb{R}^d$ denote a mesh point at index $\mathbf{l} = (l_1, \dots, l_d) \in \mathbb{N}^d$, $1 \leq l_i \leq n_i$. The coordinates \mathbf{x}_l are computed according to

$$\mathbf{x}_l = ((\mathbf{n}/2) - \mathbf{l}) \odot \mathbf{h}.$$

Here, \odot denotes an elementwise multiplication between two vectors (Hadamard product) and $\mathbf{n} = (n_1, \dots, n_d) \in \mathbb{N}^d$ represents the number of mesh points along each spatial direction. We denote the resulting mesh by $\mathbf{\Omega} = (\mathbf{x}_l) \in \mathbb{R}^{d, n_1, \dots, n_d}$. Similarly, we subdivide the unit time interval $[0, 1]$ into a uniform mesh with step size $h_t = 1/n_t$. We discretize integrals using a trapezoidal rule.

2.5.2. Time Integration. We use a SL method for numerical time integration [140]. The prototype implementation of this time integrator is described in [72]. Different variants of hardware-accelerated implementations are described in [71, 90, 91]. Other works that consider a SL scheme in a similar context are [8, 78].

The SL scheme is a hybrid between Eulerian and Lagrangian methods. It is unconditionally stable, i.e., we can select the time step size $h_t > 0$ solely based on accuracy considerations. To apply the SL scheme to the PDEs that appear in our optimality system we need to bring them into the general form

$$(14) \quad d_t u = f(t, u, \mathbf{v}, \dots)$$

where $u : [0, 1] \times \Omega \rightarrow \mathbb{R}$ denotes an arbitrary scalar function, $d_t := \partial_t + \mathbf{v} \cdot \nabla$ denotes the *material derivative*, and the right-hand-side f represents all remaining terms. To obtain this representation for the equations considered here, we use the vector calculus identity $\nabla \cdot u \mathbf{v} = u \nabla \cdot \mathbf{v} + \nabla u \cdot \mathbf{v}$.

In the first step, we have to compute the characteristic $\mathbf{y} : [t^j, t^{j+1}] \rightarrow \mathbb{R}^d$ along which particles flow between the timepoint t^j and t^{j+1} , $j = 1, \dots, n_t$. The question we seek to answer is where particles at time t^{j+1} originate from given data at time t^j . That is, we compute the departure point \mathbf{y} at time $t = t^j$. To compute this characteristic, we solve the ODE $d_t \mathbf{y} = \mathbf{v} \circ \mathbf{y}$ for $t \in [t^j, t^{j+1})$ with $\mathbf{y} = \mathbf{x}$ for $t = t^{j+1}$ backward in time. We illustrate the computation of the departure point in Figure 2.

In our implementation, we compute the characteristics \mathbf{y} using an RK2 method. Notice that the velocity \mathbf{v} is constant in time; this simplifies the computation of the transported quantities

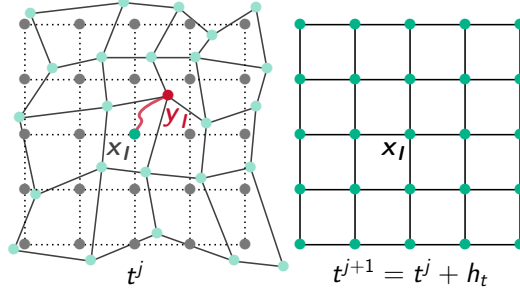


FIGURE 2. Illustration of the computation of the characteristic \mathbf{y} in the SL scheme. In the SL scheme, we have to compute the departure points at time t^j . To do so, we start with a regular grid at time t^{j+1} and solve for the characteristic \mathbf{y}_l at a given point \mathbf{x}_l at mesh index l backward in time (green line in the graphic on the left). The deformed grid configuration is overlaid onto the initial regular grid at time t^j . (Figure modified from [48, 90].)

considerably. Let $\mathbf{v}_l := \mathbf{v}(\mathbf{x}_l)$ denote the discretized velocity at a given mesh point $\mathbf{x}_l \in \Omega$. We obtain the l -th query point $\mathbf{y}_l \in \mathbb{R}^d$ associated with $\mathbf{x}_l \in \Omega$ according to

$$\begin{aligned}\tilde{\mathbf{y}}_l &\leftarrow \mathbf{x}_l - h_t \mathbf{v}(\mathbf{x}_l) \\ \mathbf{y}_l &\leftarrow \mathbf{x}_l - \frac{h_t}{2} (\mathbf{v}(\mathbf{x}_l) + \mathbf{v}(\tilde{\mathbf{y}}_l)).\end{aligned}$$

The intermediate query points $\tilde{\mathbf{y}}_l$ and the final query point \mathbf{y}_l (i.e., the departure point), are—in general—off-grid locations. Therefore, evaluating quantities of interest at these locations requires interpolation (see §2.5.4 for details). If (14) is homogeneous, i.e., $f = 0$, we only interpolate the transported quantity to obtain its value at the departure point \mathbf{y}_l at time t^j and assign the resulting value to the regular mesh point $\mathbf{x}_l \in \Omega$ at time t^{j+1} . That is,

$$u(t^{j+1}, \mathbf{x}_l) \leftarrow u(t^j, \mathbf{y}_l).$$

If (14) is not homogeneous, i.e., $f \neq 0$, we have to solve the ODE (14) along the characteristic \mathbf{y} forward in time. We do so using an RK2 scheme. That is,

$$\begin{aligned}f_0 &\leftarrow f(t^j, u(t^j, \mathbf{y}_l), \mathbf{v}(\mathbf{y}_l), \dots) \\ \tilde{u}(t^{j+1}, \mathbf{x}_l) &\leftarrow u(t^j, \mathbf{y}_l) + h_t f_0 \\ f_1 &\leftarrow f(t^{j+1}, \tilde{u}(t^{j+1}, \mathbf{x}_l), \mathbf{v}(\mathbf{x}_l), \dots) \\ u(t^{j+1}, \mathbf{x}_l) &\leftarrow u(t^j, \mathbf{y}_l) - h_t (f_0 + f_1)/2.\end{aligned}$$

Again, quantities evaluated at the query point \mathbf{y}_l at time t^j need to be interpolated. We note that these functions live on a curvilinear mesh (see Figure 2). Since we use spectral methods with a Fourier basis we cannot evaluate the differential operators that appear in f on such a mesh. As a remedy, we do not compute the derivative on this curvilinear mesh but interpolate (i.e., transport) the derivatives evaluated on a regular mesh instead.

2.5.3. Differentiation. In our past work, we considered pseudo-spectral methods with a Fourier basis for numerical differentiation [1, 30, 71, 72].

For our GPU implementation [90, 91] we have designed a mixed-precision approach to improve scalability and computational throughput. We consider 8th-order finite differences for first-order derivatives (i.e., the gradient and divergence operators). Higher order derivative operators (e.g., the Laplacian operator Δ) and their inverse are implemented using a pseudo-spectral discretization with a Fourier basis. That is, we model an arbitrary function $u : \bar{\Omega} \rightarrow \mathbb{R}$ discretized on a regular

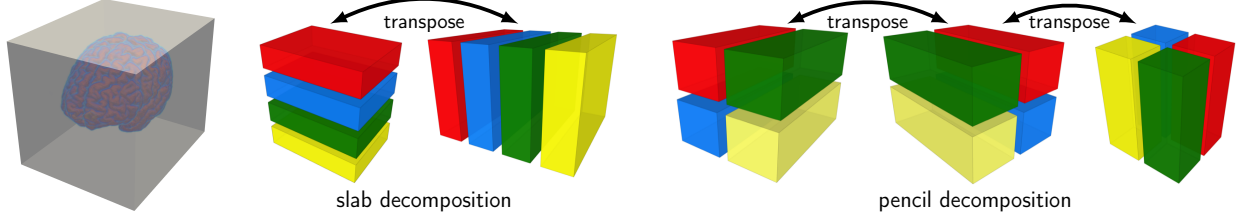


FIGURE 3. Domain decomposition for memory-distributed implementation. In each case, we assume that we use four MPI tasks to distribute our data (e.g., four GPUs or four nodes). Left: 3D volume rendering of medical imaging data set (brain image). Middle: Slab decomposition (1D domain decomposition) considered in our GPU implementation [90]. We decompose the spatial domain in the outer-most dimension. We transpose the data only once. On the right we illustrate the pencil decomposition (2D domain decomposition) of the data considered in our CPU implementation [1, 71, 93]. The computation in this data layout involves three transposes.

mesh Ω at grid points $\mathbf{x}_l \in \Omega$, $l = (l_1, \dots, l_d) \in \mathbb{N}^d$, $l_i = 1, \dots, n_i$, as $u_l := u(\mathbf{x}_l)$,

$$u_l = \sum_{k_1=-(n_1/2)+1}^{n_1/2} \cdots \sum_{k_d=-(n_d/2)+1}^{n_d/2} \hat{u}_{\mathbf{k}} \exp(-i\langle \mathbf{k}, \mathbf{x}_l \rangle_{\mathbb{R}^d})$$

with $\mathbf{k} \in \mathbb{Z}^d$ and spectral coefficients $\hat{u}_{\mathbf{k}}$. This spectral representation is the reason why we assume periodic boundary conditions in our continuous model. We note that images may not necessarily be periodic functions. We can address this by zero-padding the datasets and applying a mollifier close to the boundary $\partial\Omega$. The mapping between the coefficients $\{u_l\}$ and $\{\hat{u}_{\mathbf{k}}\}$ is done using forward and inverse FFTs. In our CPU implementation, we considered a pencil decomposition [1, 71, 93] (see Figure 3; right). Here, 1D FFTs along each spatial direction are computed based on the FFTW library. FFTs along other directions are then obtained by transposing the data, resulting in large communication costs. For the single GPU implementation described in [91] we switched to cuFFT for 3D FFTs. The multi-GPU implementation described in [90] uses a combination of cuFFT and a new 2D slab decomposition (see Figure 3; middle). This enables us to utilize the highly optimized 2D cuFFT on each GPU. We decompose the spatial domain in the outer-most dimension (i.e., x_1) and in the spectral domain in the x_2 direction. Consequently, the inner-most x_3 direction remains continuous in memory. This reduces misaligned memory access for the communication of the transpose operations. The real-to-complex transformation is divided into three steps: First, we execute cuFFT's batched 2D FFTs in the plane spanned by the x_2 and x_3 axis. Then, we transpose the complex data to a decomposition in x_2 direction. Then, we apply cuFFT's batched 1D FFTs to the x_1 direction, which is non-continuous in memory. For the inverse complex-to-real transformation, these three steps are executed in reverse order, using the respective inverse transformations. For the execution on multiple GPUs, we use CUDA-aware MPI to eliminate expensive on-node host-device transfers [90].

The 8th-order finite difference approximation of the first order derivative along the i th coordinate direction x_i at a mesh point $\mathbf{x}_l \in \Omega$ is given by

$$\partial_i u(\mathbf{x}_l) \approx \frac{u_i^-(\mathbf{x}_l) + u_i^+(\mathbf{x}_l)}{840h_i}$$

with $u_i^-(\mathbf{x}_l) := 3u_{l-4\mathbf{e}_i} - 32u_{l-3\mathbf{e}_i} + 168u_{l-2\mathbf{e}_i} - 672u_{l-\mathbf{e}_i}$, $u_i^+(\mathbf{x}_l) := 672u_{l+\mathbf{e}_i} - 168u_{l+2\mathbf{e}_i} + 32u_{l+3\mathbf{e}_i} - 3u_{l+4\mathbf{e}_i}$, and unit vectors $\mathbf{e}_i \in \{0, 1\}^d$, where the j -th entry is one for $j = i$ and zero otherwise.

The key limiting factor to obtain optimal strong and weak scalability for our method is the communication costs associated with the FFT. We refer to [90, 91] for additional details.

2.5.4. Interpolation. In our past work [90, 91], we implemented different interpolation models. To obtain the interpolated value of a function u at an arbitrary query point $\mathbf{x} = (x_1, \dots, x_d) \in \mathbb{R}^d$ we evaluate

$$u(\mathbf{x}) = \sum_{k_1=1}^{p+1} \dots \sum_{k_d=1}^{p+1} c_{k_1, \dots, k_d} \prod_{i=1}^d \phi_{k_i}(x_i),$$

where $p \in \mathbb{N}$ denotes the polynomial order, $\phi_j : \mathbb{R} \rightarrow \mathbb{R}$, $j = 1, \dots, p+1$, are the polynomial basis functions, and $c_{k_1, \dots, k_d} \in \mathbb{R}$ are the coefficients. For Lagrange interpolation, the coefficients are identical to the grid values of the discretized function u and ϕ_j are the Lagrange polynomials.

The numerical accuracy and computational performance of different variants of the interpolation kernel are discussed in [91]. The multi-GPU implementation is described in [90]. We use NVIDIA’s libraries for texture-based trilinear interpolation [141]. In [91], we also transferred our CPU kernels for cubic Lagrange interpolation [1, 33, 93] to the GPU. We implemented two variants, one that uses texture units for lookup tables and one that implements texture-based interpolation [91]. The latter implementation is similar to [142]; it yields higher computational throughput at lower accuracy. Lastly, we have developed a texture-based B-spline interpolation, the computational kernels of which are inspired by [142–144]. For the execution on multiple GPUs, we use CUDA-aware MPI to eliminate expensive on-node host-device transfers [90]. We utilize the `thrust` library [145] to determine which query points need to be processed by which GPU, thereby completely eliminating host-side computation. We use sparse point-to-point communication to send points to other processors. We adaptively allocate memory for the respective MPI buffers. We do this by computing an estimate of the maximal displacement of grid points along the computed trajectories based on the CFL number of the velocity field. We refer to [90, 91] for additional details.

2.5.5. Optimization. We use an iterative method globalized by an Armijo line search [146, 147]. The outer iterations of our algorithm are summarized in Algorithm 1 in the appendix. At (outer) iteration $k \in \mathbb{N}$, we update the iterate $\mathbf{v}^{(k)}$ according to

$$\mathbf{B}^{(k)} \tilde{\mathbf{v}}^{(k)} = -\mathbf{g}^{(k)}, \quad \mathbf{v}^{(k+1)} = \mathbf{v}^{(k)} + \gamma^{(k)} \tilde{\mathbf{v}}^{(k)},$$

where $\tilde{\mathbf{v}}^{(k)} \in \mathbb{R}^{dn}$ denotes the search direction, $\mathbf{B}^{(k)} \in \mathbb{R}^{dn, dn}$ is a positive-definite matrix, and $\gamma^{(k)} > 0$ is the step size.

For $\mathbf{B}^{(k)} = \text{diag}(1, \dots, 1) \in \mathbb{R}^{dn, dn}$ the scheme above corresponds to a gradient descent algorithm. In [30], we consider a preconditioned gradient descent algorithm. This scheme is more stable and yields an improved convergence behavior. The preconditioner is the inverse of the regularization operator \mathcal{L} . That is, $\mathbf{B}^{(k)} = \alpha \mathbf{L}$, where $\mathbf{L} \succeq \mathbf{0}$ denotes the discretization of \mathcal{L} . This scheme can be viewed as a Picard iteration. We note that our spectral discretization allows us to apply the inverse of this operator with vanishing costs; the complexity of inverting \mathbf{L} is $\mathcal{O}(n \log n)$ regardless of the Sobolev norm we consider. If the operator \mathcal{L} has a non-trivial kernel, we set the spectral coefficients that are zero to one before inverting $\alpha \mathbf{L}$. Consequently, the search direction $\tilde{\mathbf{v}}^{(k)}$ is given by $\tilde{\mathbf{v}}^{(k)} = -(\alpha \mathbf{L})^{-1} \mathbf{g}^{(k)}$.

In addition, we have designed a (Gauss)–Newton–Krylov algorithm for numerical optimization [1, 30]. Here, $\mathbf{B}^{(k)}$ corresponds to the Hessian matrix $\mathbf{H}^{(k)}$ at (outer) iteration k . Consequently, we have to invert $\mathbf{H}^{(k)}$ at each iteration. We note that forming and storing $\mathbf{H}^{(k)}$ results in prohibitive computational costs and memory requirements. As such, we cannot use direct methods [148, 149]. Instead, we use iterative methods to invert $\mathbf{H}^{(k)}$. In particular, we use matrix-free Krylov subspace methods—more precisely, a PCG algorithm [150]—to compute the action of the inverse of $\mathbf{H}^{(k)}$ on the vector $-\mathbf{g}^{(k)}$. As such, we only require an expression for the Hessian matvec. This is precisely what is given by (11). Thus, we need to evaluate (11) at every inner iteration of our Krylov-subspace method. This involves solving the PDEs (12) and (13) at every inner iteration

of the PCG algorithm. These matvecs constitute the most expensive part of our algorithm. We summarize this algorithm in Algorithm 2 in the appendix.

We note that we can use other iterative methods to compute the action of the inverse of the Hessian. In fact, we have tested different methods. In our experiments, we did not observe any issues with the PCG algorithm nor did we see any benefits from using different iterative methods. Since the Hessian is (also for all practical purposes, in computation) a symmetric positive definite operator we prefer to use the PCG method over, e.g., GMRES. Having said this, we note that CLAIRE supports different Krylov subspace methods via PETSc [115, 116]. We discuss this in greater detail in the next subsection.

Since the considered optimization problem is, in general, non-convex, one additional challenge that arises is that the Hessian is not guaranteed to be positive definite, especially far away from a (local) minimizer. One approach to address this issue is to terminate the PCG algorithm as soon as one detects negative curvature. In this case, we use the former iterate of the PCG algorithm as a search direction. We consider a Gauss–Newton approximation to $\mathbf{H}^{(k)}$ [1, 30, 48] as an alternative to this approach. This approximation is guaranteed to be positive semi-definite. On the downside, we can expect the convergence to drop from quadratic to superlinear. This Gauss–Newton approximation is obtained by dropping all terms that involve the dual variable λ in (11) and (13), respectively. Notice that the final condition for the dual variable λ in (10) corresponds to the mismatch between the transported intensities of the template image m_0 and the reference image m_1 . Thus, as we approach a (local) minimizer of our problem, we can expect that λ tends to zero; our Gauss–Newton approximation becomes exact and we recover quadratic convergence.

To further amortize computational costs, we do not invert $\mathbf{H}^{(k)}$ exactly. We consider an inexact scheme [146, 151, 152]. This is accomplished by selecting the stopping condition for the PCG method to be proportional to the norm of the reduced gradient; as we approach a (local) minimizer, the tolerance decreases and we solve for the search direction more accurately. That is, we terminate the algorithm if

$$\|\mathbf{r}^{(k)}\|_\infty \leq \eta^{(k)} \|\mathbf{g}^{(k)}\|_\infty, \quad \mathbf{r}^{(k)} := \mathbf{H}^{(k)} \tilde{\mathbf{v}}^{(k)} + \mathbf{g}^{(k)},$$

with forcing sequence $\eta^{(k)} = \min(1/2, \sqrt{\|\mathbf{g}^{(k)}\|_\infty})$ or $\eta^{(k)} = \min(1/2, \|\mathbf{g}^{(k)}\|_\infty)$ for superlinear or quadratic convergence, respectively. See Algorithm 2, line 13 in the appendix.

In [30] we demonstrate that the preconditioned gradient descent scheme is less effective than our (Gauss–)Newton–Krylov scheme. As such, we only consider our (Gauss–)Newton–Krylov algorithm here.

We terminate the optimization if we reduce the gradient by $\epsilon_{\text{opt}} > 0$, i.e.,

$$\|\mathbf{g}^{(k)}\|_\infty \leq \epsilon_{\text{opt}} \|\mathbf{g}^{(0)}\|_\infty$$

or if $\|\mathbf{g}^{(k)}\|_\infty \leq 1\text{e-}6$. We have implemented alternative stopping criteria [30] but do not consider them here.

2.5.6. Preconditioning. The main cost of the (Gauss–)Newton–Krylov algorithm is the solution of the linear system

$$(15) \quad \mathbf{H}^{(k)} \tilde{\mathbf{v}}^{(k)} = -\mathbf{g}^{(k)}, \quad k = 1, 2, 3, \dots,$$

at each outer iteration k , with $\mathbf{H}^{(k)} = \mathbf{H}_{\text{reg}}^{(k)} + \mathbf{H}_{\text{dat}}^{(k)}$, where $\mathbf{H}_{\text{reg}}^{(k)} = \mathbf{H}_{\text{reg}} \in \mathbb{R}^{dn, dn}$ is a discrete representation of the regularization operator $\mathcal{H}_{\text{reg}} = \alpha \mathcal{L}$ and $\mathbf{H}_{\text{dat}}^{(k)} \in \mathbb{R}^{dn, dn}$ is the discrete version of \mathcal{H}_{dat} in (11). For the model outlined in §2.2 the Hessian behaves like a compact operator; large eigenvalues are associated with smooth eigenvectors and the eigenvectors become more oscillatory as the eigenvalues decrease [30].

To amortize the computational costs of our algorithm and make it competitive with gradient descent schemes that consider first-order derivative information only, we have to design effective methods for preconditioning the linear system given above. That is, we seek a matrix $\mathbf{M}^{(k)} \succ \mathbf{0}$

such that, ideally, $(\mathbf{M}^{(k)})^{-1}\mathbf{H}^{(k)} \approx \mathbf{I}_{dn}$, $\mathbf{I}_{dn} := \text{diag}(1, \dots, 1) \in \mathbb{R}^{dn, dn}$. This makes approximations to $\mathbf{H}^{(k)}$ (that are “easy” to invert) an obvious choice.

Regularization Preconditioner. A common choice in PDE-constrained optimization is to consider the regularization operator \mathbf{H}_{reg} as a preconditioner \mathbf{M} [153–155]. The preconditioned Hessian is a perturbation of the identity, i.e.,

$$(\mathbf{H}_{\text{reg}})^{-1}\mathbf{H}^{(k)} = \mathbf{I}_{dn} + (\mathbf{H}_{\text{reg}})^{-1}\mathbf{H}_{\text{dat}}^{(k)}$$

with $\mathbf{I}_{dn} := \text{diag}(1, \dots, 1) \in \mathbb{R}^{dn, dn}$. Since $\mathbf{H}_{\text{reg}} \succeq \mathbf{0}$ is a (high-order) differential operator (typically, a Helmholtz type operator), its inverse acts as a smoother. We note that applying the inverse of \mathbf{H}_{reg} has a complexity of $\mathcal{O}(n \log n)$ in our implementation, i.e., we have to compute two FFTs and a diagonal scaling in the spectral domain using the appropriate Fourier coefficients. As such, this strategy for preconditioning the reduced space Hessian has vanishing costs. This preconditioner has been considered in [15, 30, 33, 71]. The performance of this preconditioner is mesh independent (assuming we can entirely resolve the problem on the coarsest mesh). However, it deteriorates significantly as we decrease the regularization parameter α .

Two-Level Preconditioner. Inspired by multi-grid approaches, we designed a two-level preconditioner for the reduced space Hessian [1, 48]. We use a coarse grid approximation of the inverse of the reduced space Hessian as a preconditioner. The basic idea is to iterate only on the low-frequency part and ignore the high-frequency components. That is, we use the inverse of the reduced space Hessian $\mathbf{H}^{(k)}$, inverted on a coarser grid, as a preconditioner. This idea is motivated by the work in [156–161]. For simplicity of notation, we drop the dependence of the Hessian on the outer iteration index k .

We decompose the Hessian into two operators \mathbf{H}_L and \mathbf{H}_H —one acting on low and the other acting on high frequencies, respectively. We denote the operators that project on the low and high-frequency subspaces by $\mathbf{P}_L : \mathbb{R}^{dn} \rightarrow \mathbb{R}^{dn}$ and $\mathbf{P}_H : \mathbb{R}^{dn} \rightarrow \mathbb{R}^{dn}$. Let $\mathbf{e}_j \in \mathbb{R}^n$, $(\mathbf{e}_j)_i = 1$ if $j = i$ and $(\mathbf{e}_j)_i = 0$ for $i \neq j$, $i, j = 1, \dots, n$, denote an eigenvector of \mathbf{H} with $(\mathbf{P}_L \mathbf{H} \mathbf{P}_H) \mathbf{e}_j = (\mathbf{P}_H \mathbf{H} \mathbf{P}_L) \mathbf{e}_j = \mathbf{0}$. Then, with $\mathbf{P}_H + \mathbf{P}_L = \mathbf{I}_{dn}$, $\mathbf{I}_{dn} := \text{diag}(1, \dots, 1) \in \mathbb{R}^{dn, dn}$, we have

$$\mathbf{H} \mathbf{e}_j = (\mathbf{P}_H + \mathbf{P}_L) \mathbf{H} (\mathbf{P}_H + \mathbf{P}_L) \mathbf{e}_j = \mathbf{P}_H \mathbf{H} \mathbf{P}_H \mathbf{e}_j + \mathbf{P}_L \mathbf{H} \mathbf{P}_L \mathbf{e}_j,$$

In general, this equality will not hold. However, we are not interested in using this model as a surrogate for the Hessian \mathbf{H} ; we are merely interested in designing an effective preconditioner \mathbf{M} so that $\text{cond}(\mathbf{M}^{-1}\mathbf{H}) \ll \text{cond}(\mathbf{H})$.

Suppose we can decompose $\tilde{\mathbf{v}} \in \mathbb{R}^{dn}$ into a smooth component $\tilde{\mathbf{v}}_L \in \mathbb{R}^{dn}$ and a high-frequency component $\tilde{\mathbf{v}}_H \in \mathbb{R}^{dn}$, where each of these vectors can be found by solving

$$\mathbf{H}_L \tilde{\mathbf{v}}_L = (\mathbf{P}_L \mathbf{H} \mathbf{P}_L) \tilde{\mathbf{v}}_L = -\mathbf{P}_L \mathbf{g} \quad \text{and} \quad \mathbf{H}_H \tilde{\mathbf{v}}_H = (\mathbf{P}_H \mathbf{H} \mathbf{P}_H) \tilde{\mathbf{v}}_H = -\mathbf{P}_H \mathbf{g},$$

respectively. We use this construction to design an effective preconditioner for the smooth spectrum of our problem. Let $\mathbf{r} \in \mathbb{R}^{dn}$ denote the vector we apply the inverse of our preconditioner $\mathbf{M} \in \mathbb{R}^{dn, dn}$ to. Since our implementation is matrix-free, we iteratively solve $\mathbf{M} \mathbf{s} = \mathbf{r}$ to obtain the action of the inverse of \mathbf{M} on \mathbf{r} . In the spirit of the conceptual idea introduced above, we assume that we can decompose \mathbf{s} into a smooth component \mathbf{s}_L and a high-frequency component \mathbf{s}_H . Let $\mathbf{Q}_R \in \mathbb{R}^{dn/2, dn}$ denote a restriction operator and $\mathbf{Q}_P \in \mathbb{R}^{dn, dn/2}$ denote prolongation operator. Moreover, let $\mathbf{F}_L \in \mathbb{R}^{dn, dn}$ and $\mathbf{F}_H \in \mathbb{R}^{dn, dn}$ denote a low and high-pass filter, respectively. We project the vector $\mathbf{r} \in \mathbb{R}^{dn}$ to a vector $\mathbf{r}_L \in \mathbb{R}^{dn/2}$ by filtering the high-frequency components and restricting the resulting vector to a coarser mesh, i.e., $\mathbf{r}_L = \mathbf{Q}_R \mathbf{F}_L \mathbf{r}$. Subsequently, we obtain the smooth component \mathbf{s}_L by solving

$$\tilde{\mathbf{M}}_L \tilde{\mathbf{s}}_L = \mathbf{Q}_R \mathbf{F}_L \mathbf{r}$$

where $\tilde{\mathbf{M}}_L \in \mathbb{R}^{dn/2, dn/2}$ is a coarse grid approximation of the low-frequency part of the reduced space Hessian \mathbf{H} and $\tilde{\mathbf{s}}_L \in \mathbb{R}^{dn/2}$. This allows us to precondition the smooth part of \mathbf{r} . We note that we do not precondition the high-frequency components $\mathbf{r}_H := \mathbf{F}_H \mathbf{r}$ of \mathbf{r} , where $\mathbf{F}_H \in \mathbb{R}^{dn, dn}$

is a high-pass filter with $\mathbf{F}_H + \mathbf{F}_L = \mathbf{I}_{dn}$. Consequently, $\mathbf{s}_H = \mathbf{F}_H \mathbf{r}$. In summary, the solution of $\mathbf{M}\mathbf{s} = \mathbf{r}$ is given by

$$\mathbf{s} = \mathbf{s}_L + \mathbf{s}_H \approx \mathbf{Q}_P \mathbf{F}_L \tilde{\mathbf{s}}_L + \mathbf{F}_H \mathbf{r} \approx \mathbf{Q}_P \mathbf{F}_L (\tilde{\mathbf{M}}_L)^{-1} \mathbf{Q}_R \mathbf{F}_L \mathbf{r} + \mathbf{F}_H \mathbf{r}.$$

To counter the fact that we leave the high-frequency components untouched, we do not directly apply this preconditioner to the reduced-space KKT system in (15) but the regularization preconditioned system

$$(\mathbf{I} + \mathbf{H}_{\text{reg}}^{-1/2} \mathbf{H}_{\text{data}} \mathbf{H}_{\text{reg}}^{-1/2}) \mathbf{w} = -\mathbf{H}_{\text{reg}}^{-1/2} \mathbf{g}$$

where $\mathbf{w} := \mathbf{H}_{\text{reg}}^{1/2} \tilde{\mathbf{v}}$. Notice that the square root of the inverse of \mathbf{H}_{reg} acts as a smoother. This scheme can be viewed as an approximation of a two-level multigrid V-cycle with an explicit (algebraic) smoother $\mathbf{H}_{\text{reg}}^{-1/2}$.

Before we explore extensions of this idea, we present some implementation aspects. We use spectral restriction and prolongation operators \mathbf{Q}_R and \mathbf{Q}_P [1, 48]. The operators \mathbf{F}_L and \mathbf{F}_H are implemented as cut-off filters in the frequency domain [1, 48]. For the implementation of the coarse grid operator $\tilde{\mathbf{M}}_L$ we have two choices. First, we can use a Galerkin discretization, which is formally given by $\tilde{\mathbf{M}}_L = \mathbf{Q}_R \mathbf{H} \mathbf{Q}_P$ [162]. The drawback of this approach is that we do not significantly reduce the computational costs compared to inverting the fine-grid Hessian, since each matvec necessitates the solution of the incremental state and adjoint equation at full resolution. Conversely, we can directly discretize the Hessian on a coarse grid to obtain $\tilde{\mathbf{M}}_L$. This makes the implementation slightly more involved but reduces the computational costs drastically. We opt for the latter approach [1, 48].

To invert the matrix $\tilde{\mathbf{M}}_L$ we have several options. Again, traditional direct methods are out of the question. However, we can use a nested Krylov-subspace method. If we use a Krylov-subspace method as an outer method (i.e., for computing the search direction), we have to select a tolerance for the inner Krylov-subspace method that is a fraction of the tolerance used to compute the search direction. Alternatively, we can replace the solver for the Newton step with a flexible Krylov-subspace method [163, 164] and use a fixed number of iterations for the nested (inner) Krylov-subspace method. Alternatively, we can use a semi-iterative Chebyshev method [165] with a fixed number of iterations on the inside. This yields a fixed linear operator for a particular choice of eigenvalue bounds [166]. These bounds can be estimated using a Lanczos method. We have tested and compared these approaches in [1, 48]. This also includes the use of different Krylov-subspace methods for not only applying the preconditioner but also solving for the Newton step such as the standard and flexible GMRES method, the standard and flexible PCG method, or the Chebyshev method (some of which have been mentioned above). In [1], we observed that the nested PCG method converges most quickly in the 3D setting.

Zero Velocity Approximation. The preconditioner introduced in the former section requires a repeated evaluation of the incremental state and adjoint equations. The savings come from discretizing the reduced space Hessian on a mesh of half the resolution. In [90] we developed a preconditioner that does not require solving any PDEs; the Hessian operator is fixed across all iterations. This is accomplished by fixing \mathbf{v} to $\mathbf{v} = \mathbf{0}$ (our initial guess for the optimization problem). Under the assumption, the state equation simplifies to $\partial_t m = 0$, i.e., $m(t, \mathbf{x}) = m_0(\mathbf{x})$ for all $\mathbf{x} \in \Omega$ and $t \in [0, 1]$. Likewise, we have $\partial_t \lambda = 0$, i.e., $\lambda(t, \mathbf{x}) = -(m_0(\mathbf{x}) - m_1(\mathbf{x}))$ for all $\mathbf{x} \in \Omega$ and $t \in [0, 1]$. Inserting these expressions into the incremental state equation we have $\partial_t \tilde{m} = -\nabla m_0 \cdot \tilde{\mathbf{v}}$, which implies that $\tilde{m}(t=1) = -\nabla m_0 \cdot \tilde{\mathbf{v}}$. The incremental adjoint equation for the Gauss–Newton approximation for $\mathbf{v} = \mathbf{0}$ is given by $\partial_t \tilde{\lambda} = 0$, i.e., $\tilde{\lambda}(t, \mathbf{x}) = \nabla m_0(\mathbf{x}) \cdot \tilde{\mathbf{v}}(\mathbf{x})$ for all $\mathbf{x} \in \Omega$ and $t \in [0, 1]$. Consequently, the Gauss–Newton approximation of the Hessian matvec for $\mathbf{v} = \mathbf{0}$ is given by

$$\mathcal{H}_0[\tilde{\mathbf{v}}](\mathbf{x}) = \alpha \mathcal{L} \tilde{\mathbf{v}}(\mathbf{x}) + (\nabla m_0(\mathbf{x}) \otimes \nabla m_0(\mathbf{x})) \tilde{\mathbf{v}}(\mathbf{x}).$$

This approximation deteriorates as we move away from our initial guess $\mathbf{v}^{(0)} = \mathbf{0}$. As a remedy, we replace m_0 in the expression above with our current estimate m at $t = 1$ at each outer iteration k for a trial velocity $\mathbf{v}^{(k)}$. Like in previous sections, we do not form or store $\mathbf{H}_0 \in \mathbb{R}^{dn, dn}$ (the discrete version of \mathcal{H}_0); we invert the matrix iteratively using a nested PCG method. To further reduce the computational costs, we combine the \mathbf{H}_0 approximation with the two-level scheme discussed above. That is, we replace the coarse grid preconditioner \mathbf{M}_L with a coarse grid approximation of \mathcal{H}_0 .

2.6. Parameter Selection. Based on empirical observations, we fix most of our numerical parameters. For the number of time steps n_t in the numerical time integration we found that $n_t = 4$ provides sufficient accuracy to obtain a good matching between images at resolutions at the order of 256^3 (standard size for brain images acquired in clinical practice). We set the tolerance for the relative reduction of the gradient (stopping condition for optimization) to $\epsilon_{\text{opt}} = 5\text{e-}2$. We use a superlinear forcing sequence to compute the tolerance for the outer PCG algorithm. We use a two-level implementation of the zero velocity approximation of the reduced space Hessian as a preconditioner. The tolerance for the inner PCG to invert \mathbf{H}_0 is 10 times smaller than the outer tolerance of the PCG. The formulation we consider for diffeomorphic image registration is an extension of what we discussed so far; it considers near-incompressible velocities. We describe this formulation in greater detail in the appendix. The regularization parameter for the H^1 penalty for the divergence of the velocity field is fixed and set to $\beta = 1\text{e-}4$. We compute an optimal regularization parameter α as described below.

Several methods exist to estimate an optimal regularization parameter for inverse problems (see, e.g., [167] for examples). All of these methods have in common that the estimation of an optimal regularization parameter is expensive. Methods that assume that the differences between model output and observed data are random (such as, e.g., generalized cross validation) are not necessarily reliable in the context of image registration, since imaging noise is prone to be highly structured [168]. In our work, we consider a binary search for identifying an optimal value α for the regularization model for the velocity field \mathbf{v} [1, 30, 112]. This approach is in spirit similar to an L-curve strategy. Related parameter continuation strategies have been considered in [68, 168, 169]. As a measure for optimality, we select bounds on the determinant of the deformation gradient $\det \nabla \mathbf{y}$. Notice that we do not compute \mathbf{y} to obtain this quantity but solve a transport problem (see appendix). Assuming that we start from an identity map $\text{id}_{\mathbb{R}^d}$ the initial value for $\det \nabla \mathbf{y}$ is one (this is equivalent to $\mathbf{v} = \mathbf{0}$ in our formulation). Consequently, we assume that the map is diffeomorphic if $\det \nabla \mathbf{y} \geq 0$. This motivates the use of a lower bound $\epsilon_D > 0$. Since the determinant of the deformation gradient of \mathbf{y} is inversely proportional to $\det \nabla \mathbf{y}^{-1}$, we use $1/\epsilon_D$ as an upper bound. Consequently, we require

$$(16) \quad \epsilon_D < \det \nabla \mathbf{y} < 1/\epsilon_D$$

for any admissible \mathbf{y} . Our approach is as follows: We start with a regularization parameter of $\alpha = 1$ and reduce α by one order of magnitude until the condition in (16) is violated. Subsequently, we perform a binary search in the interval between the last value α for which (16) held and the value for which (16) was violated. For each new trial parameter $\alpha^{(l)}$ at level $l \in \mathbb{N}$, we use the control variable $\mathbf{v}_\alpha^{(l-1)} := \mathbf{v}(\alpha^{(l-1)})$ obtained for $\alpha^{(l-1)}$ at the prior level $l - 1$ as an initial guess to speed up convergence. More details can be found in [1, 30].

Obviously, this search is expensive since we have to solve the inverse problem for each trial $\alpha^{(l)}$, $l = 0, 1, 2, 3, \dots$. Once we have identified an adequate regularization parameter α^* for a particular application, we perform a parameter continuation to speed up convergence. That is, we solve the inverse problem consecutively for different values for α , starting with $\alpha^{(0)} = 1$ and subsequently reducing $\alpha^{(l)}$ by one order of magnitude until we reach the order of α^* . Then, we solve the problem one last time for α^* . For high regularization parameters $\alpha^{(l)}$ we essentially solve a convex problem; we expect quick convergence to a (local) minimizer. We use the estimate for the

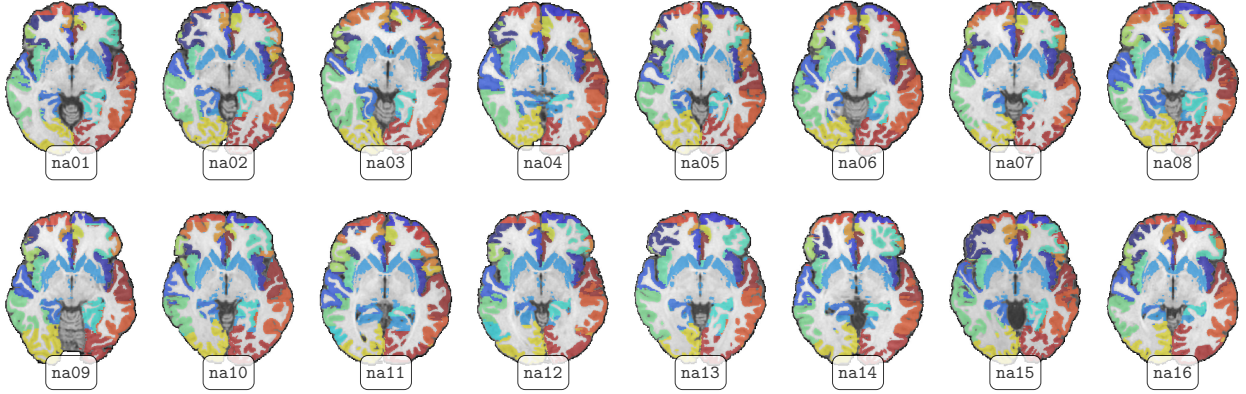


FIGURE 4. NIREP data repository [172]. We show an axial slice of each dataset (slice number 128). The repository contains 16 rigidly aligned T1-weighted MRI brain datasets (na01–na16) of size $256 \times 300 \times 256$ voxels of different individuals. Each dataset is equipped with 32 labels of anatomical gray matter regions. We overlay these regions in different colors on the MRI data. We refer to [172] for additional information about the datasets, the imaging protocol, and the preprocessing.

control variable $\mathbf{v}_\alpha^{(l-1)}$ as an initial guess for the next solve at level l . This does not significantly affect the runtime compared to directly solving our problem for α^* . Moreover, it “convexifies” the problem; we anticipate to more quickly converge to a “better” (local) minimizer and/or avoid getting trapped in “less optimal” local minima. We have compared this strategy against multi-scale (scale continuation) and multi-resolution (grid continuation) approaches in [1]. We observed the parameter continuation to be more stable and overall more effective. Combining parameter continuation with scale and/or grid continuation remains subject to future work.

Lastly, we note that machine learning has also recently been considered for regularization operator and parameter tuning [170, 171].

3. RESULTS

We consider a slightly more involved formulation than the one presented in §2.2. We provide additional details in the appendix. We refer to [1, 33, 90, 93] for weak and strong scaling results of our CPU and GPU implementation of CLAIRE, respectively. In the present work, we limit the performance analysis to a single GPU.

3.1. Data. We report results for the NIREP dataset [172]. We refer to [172] for additional information about the datasets, the imaging protocol, and the preprocessing. This repository contains 16 rigidly aligned T1-weighted MRI brain datasets (na01–na16) of size $256 \times 300 \times 256$ voxels of different individuals. Consequently, we invert for $3(256^2)(300) = 58,982,400$ unknowns. Each dataset is equipped with 33 labels for anatomical gray matter regions. These labels allow us to assess the performance of the registration; we assess registration accuracy by how well these labels are mapped to one another. To do so, we compute the so-called Dice between the label maps. For a Dice of one, the labels are in perfect agreement. For a Dice of zero, they do not overlap. Notice that the registration software does not consider these labels; registration is solely based on matching corresponding image intensities. That is, we do not explicitly minimize the alignment of the labels but only the mismatch between the data. We show the considered data in Figure 4. In particular, we show axial slices of all 16 datasets with the associated labels in the overlay.

3.2. Preconditioning. We show representative results for the convergence of different preconditioners in Figure 5. We consider the regularization preconditioner as well as two variants of the zero-velocity preconditioner—inverting the zero-velocity approximation of the reduced space

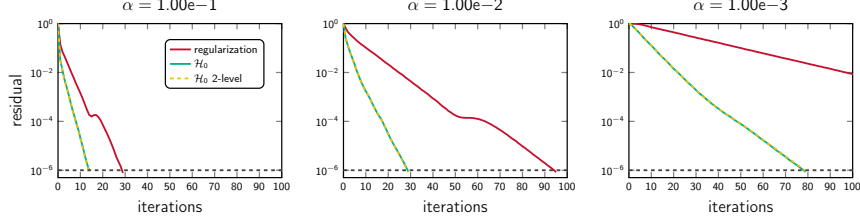


FIGURE 5. Convergence of the PCG method for solving for the Newton step. We solve for the search direction at the solution of the registration problem (dataset **na02** registered to **na01**). We consider the squared L^2 -distance. We solve this problem at the original resolution of the data. We consider three different preconditioners: The regularization preconditioner, the zero-velocity approximation of the reduced space Hessian, and the 2-level implementation of the zero-velocity approximation of the reduced space Hessian. We solve the problem for three different regularization parameter values (from left to right): $\alpha = 1e-1$, $\alpha = 1e-2$, and $\alpha = 1e-3$. The tolerance for the PCG method is $1e-6$. We plot the relative residual.

Hessian on the fine mesh and a two-level implementation of this preconditioner. To test the performance, we invert the reduced space Hessian at the true solution of our problem. That is, we solve the registration problem between two images (dataset **na02** registered to **na01**) in our case. We then use the obtained velocity as iterate at which we compute the search direction. We set the tolerance for the PCG method to $1e-6$. We consider a squared L^2 -distance as a similarity measure. We report results for the full resolution, only; $\mathbf{n}_x = (256, 300, 256)$. We report convergence results for three different choices of α ; $\alpha = 1e-1$, $\alpha = 1e-2$, and $\alpha = 1e-3$, respectively.

The most important observations are: (i) The convergence of all methods is sensitive with respect to the choice of α . (ii) The zero-velocity approximation yields an improved rate of convergence. (iii) For the zero-velocity approximation, the convergence does not deteriorate as we switch from full resolution to a coarse resolution (2-level implementation).

3.3. Regularization Parameter Search. We set the regularization parameter for the divergence of the velocity to $\beta = 1e-5$ and search for an optimal regularization parameter α using the scheme described in §2.6. We register each image with all other images. We also perform the reverse registration. This results in a total of $16(15) = 240$ registrations. We consider a squared L^2 -distance for the similarity measure.

We illustrate the search for an optimal regularization parameter for two registration problems (**na01** to **na14** and **na14** to **na01**) in Figure 6. We show representative registration results for two images from the considered NIREP dataset in Figure 7. We report statistics for the estimated regularization parameter α in Figure 8 (left plot). We also compute the minimal, mean, and maximum value of the determinant of the deformation gradient for all registrations. We report the statistics across all 240 registrations for these in Figure 8 (plots to the right). For the minimum value of the determinant of the deformation gradient, we obtained $1.668e-1$ with a standard deviation of $6.119e-2$, an overall lowest minimum value of $1.001e-1$ and an overall largest minimum value of $4.192e-1$. For the mean value of the determinant of the deformation gradient, we obtained $1.028e0$ with a standard deviation of $1.183e-2$, an overall lowest minimum value of $1.008e0$ and an overall largest minimum value of $1.084e0$. For the maximum value of the determinant of the deformation gradient, we obtained $7.150e0$ with a standard deviation of $2.324e0$, an overall lowest minimum value of $2.269e0$ and an overall largest minimum value of $9.993e0$. We report the workload for this search in Table 2.

The most important observations are: (i) We can efficiently determine an adequate regularization parameter with an average runtime of 15.135 seconds (standard deviation: 11.323 seconds), (ii) the computed deformation maps are diffeomorphic (up to numerical accuracy), and (iii) we overall obtain high-quality registration results with precise control on the determinant of the deformation gradient.

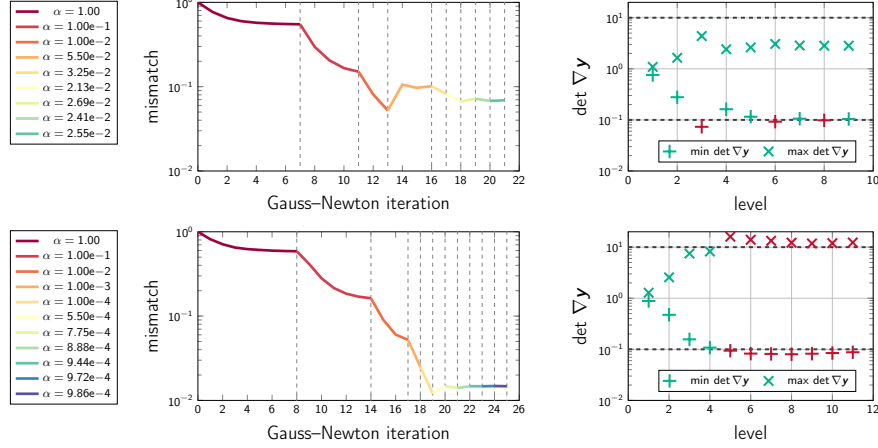


FIGURE 6. Illustration of the parameter search for the registration of the dataset with id **na14** to the dataset with id **na01** (top panel) and in the reverse direction (bottom panel). The registration results are shown in Figure 7 (bottom panel). We show (for each case) the trend of the mismatch for each choice of regularization parameter α (left) and the largest and smallest value of the determinant of the deformation gradient. In the latter plot we also show the lower and upper bound of 0.1 and 10, respectively, for the determinant of the deformation gradient as a dashed line. Whenever these bounds are violated, the marker switches from “green” to “red.”. For the run shown in the top panel, the optimal regularization parameter is $2.547\text{e-}2$. For the run at the bottom, the optimal regularization parameter is $1\text{e-}3$.

TABLE 2. Workload for estimating the regularization parameter α . We consider a squared L^2 -distance measure. All reported numbers are computed across all 240 registrations. We report the number of outer iterations, the number of Hessian matvecs, the number of PDE solves, the relative mismatch after registration, the relative change of the norm of the gradient, and the runtime (in seconds). These numbers are for solving the inverse problem multiple times; we search for an optimal regularization parameter using a binary search.

	mean	stdev	min	max	median	1st QT	3rd QT
iterations	20.538	2.268	17	28	20	19	22
matvecs	166.971	124.749	39	388	92	64	326
PDE solves	447.975	318.877	134	1008	252	190	867
mismatch	4.508e-2	3.993e-2	4.422e-3	1.756e-1	3.183e-2	1.000e-2	7.095e-2
gradient	1.394e-2	5.733e-3	3.166e-3	3.350e-2	1.338e-2	8.995e-3	1.836e-2
runtime	15.135	11.323	3.974	37.867	8.125	5.877	29.232

3.4. Registration Accuracy. In this section we assess the registration accuracy. In particular, we report the Dice values for the parameter search described in the former section. Aside from considering a squared L^2 -distance we also report registration accuracy for normalized cross correlation as a similarity measure (see appendix for details). In Figure 9 we report the Dice score for the individual labels. We report the statistics for the 240 registration runs in Table 3. Here, we compute the union of all 33 labels and report the global Dice score. We report additional results in the appendix.

The most important observations are: (i) CLAIRE yields an excellent agreement for the overall Dice with an increase from 0.551 (standard deviation: 0.041) before registration to 0.831 (standard deviation: 0.059) for the squared L^2 -distance and 0.835 (standard deviation: 0.060) for normalized cross correlation. (ii) The performance for the squared L^2 -distance and normalized cross correlation are en par for our current implementation.

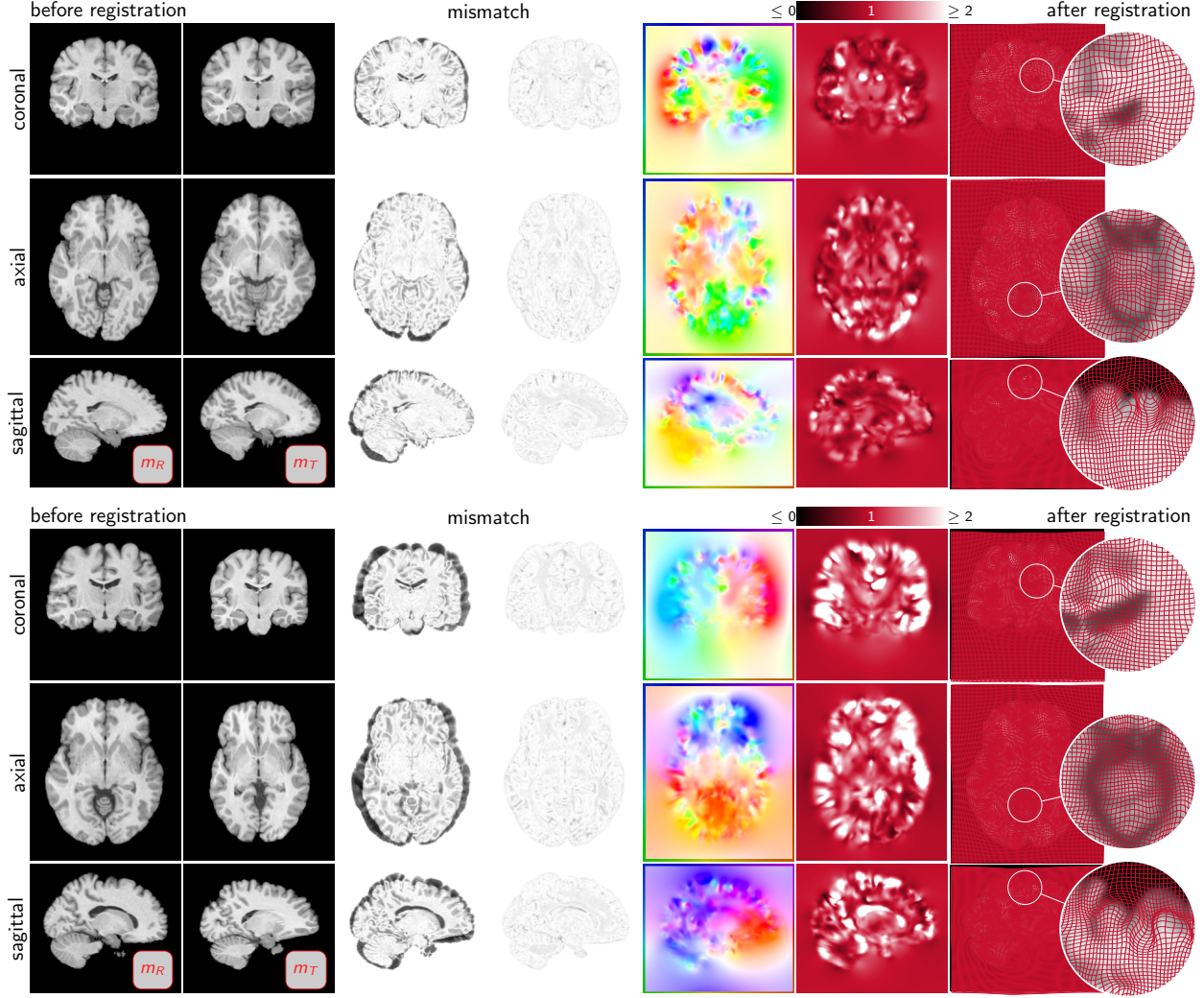


FIGURE 7. Representative registration results for CLAIRE. We consider an H^1 -seminorm as a regularization model for the velocity field and an H^1 -norm to regularize the divergence of the velocity field. We model near incompressible flows. The regularization parameter for the divergence is set to $\beta = 1e-5$. The regularization for the velocity is estimated. The bound for the determinant of the deformation gradient is set to $1e-1$. We register the dataset with id **na06** to the dataset with id **na02** (top panel) and the dataset with id **na14** to the dataset with id **na01** (bottom panel) of the NIREP repository. The data is rigidly aligned. For each panel, we show the following: The top row shows the coronal view, the middle row the axial view, and the bottom row the sagittal view of the 3D volume. The columns are (from left to right) (i) the template image m_0 , (ii) the reference image m_1 , (iii) the residual differences between the reference image and the template image (before registration; large differences are colored in black and no residual difference are colored in white), (iv) the residual differences between the deformed template image and the reference image (after registration), (v) an illustration of the velocity field (color represents orientation; see boundary), (vi) visualization of the determinant of the deformation gradient (color bar on top), (vii) and an illustration of the projection of the computed deformation map onto the corresponding plane.

3.5. Convergence and Runtime. In the former section, we have seen how CLAIRE performs when searching for an optimal regularization parameter for each individual volume. In the current section, we fix the regularization parameter to the mean optimal value of $\alpha = 1.773e-3$ determined in the former section and focus on computational performance. We plot the residual vs. the number of outer iterations in Figure 10. Here, we solve the inverse problem for a fixed $\alpha = 1.773e-3$ without

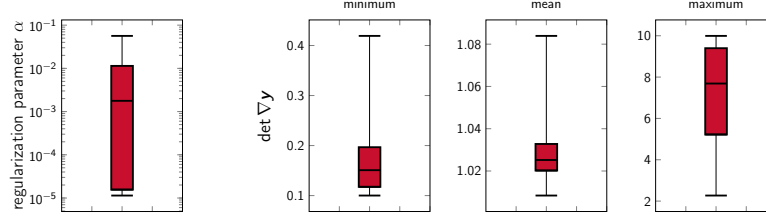


FIGURE 8. Statistics for the estimation of the regularization parameter α across 240 registrations between all NIREP datasets. The data has been rigidly registered. We report the estimated regularization parameters α (right) and the statistics for the minimum, mean and maximum of the determinant of the deformation gradient across each individual registration. The estimated regularization parameter α is $7.525\text{e-}3$ with a standard deviation of $1.098\text{e-}2$, a median of $1.773\text{e-}3$, a minimal value of $1.141\text{e-}5$, and a maximal value of $5.641\text{e-}2$ across all 240 registrations. The 25th percentile is $1.563\text{e-}5$ and the 75th percentile is $1.141\text{e-}2$. The minimal, mean and maximal values are $1.668\text{e-}1$ (standard deviation: $6.119\text{e-}2$), $1.028\text{e}0$ (standard deviation: $1.183\text{e-}2$), and $7.150\text{e}0$ (standard deviation: $2.324\text{e}0$).

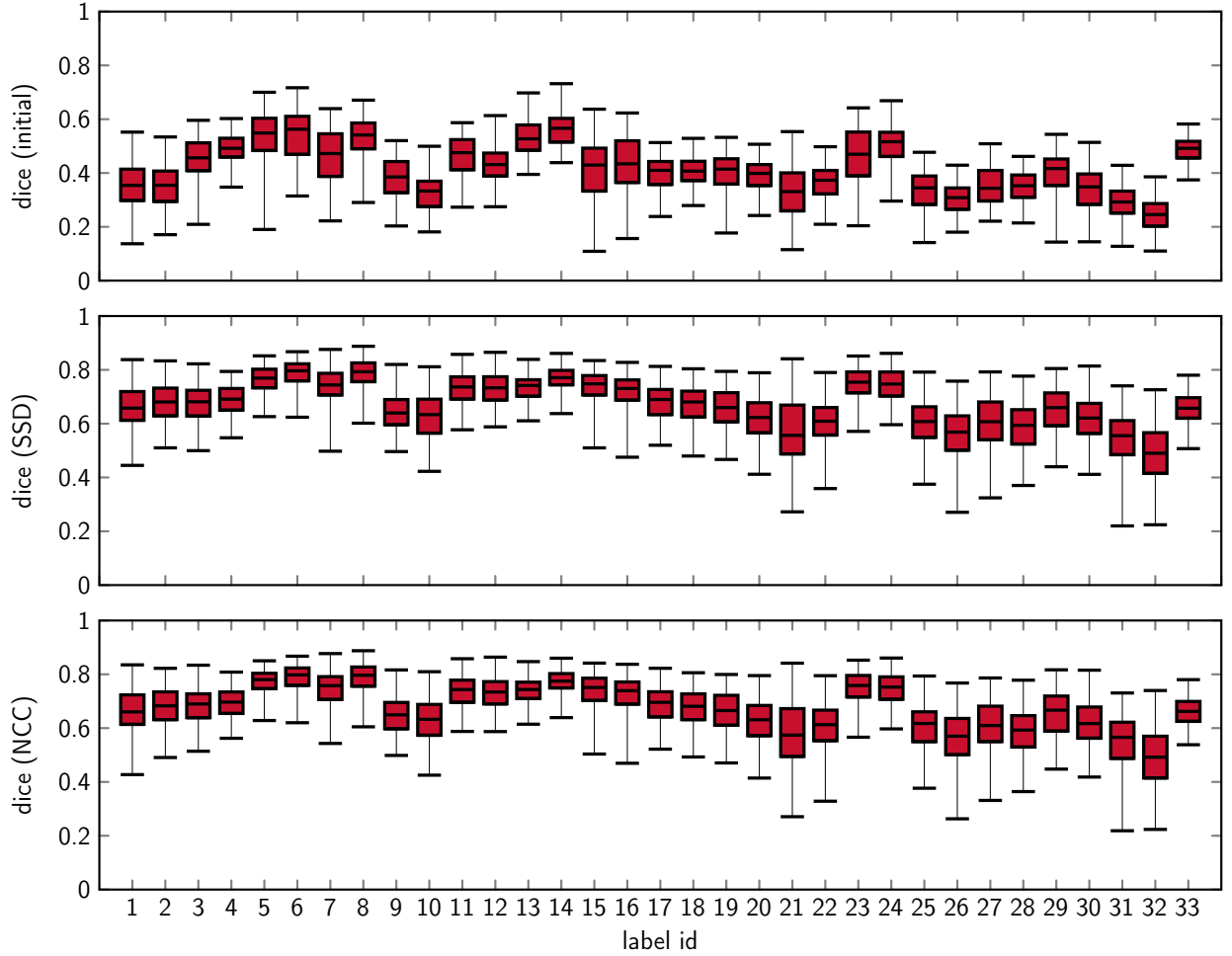


FIGURE 9. Average Dice score for individual labels. We show box plots for the Dice score for each individual label. The statistics are computed for all 240 registrations. The top row corresponds to the Dice values before registration. The middle row shows values for the Dice score after registration using a squared L^2 -distance as a similarity measure. The bottom row shows the results obtained for normalized cross correlation. We report statistics for the union of these labels in Table 3.

TABLE 3. Average DICE values. We report the mean, min, max, and median value as well as the 1st quantile and the 3rd quantile. These values are computed for the union of all labels. We report the initial values in the first row. The values after diffeomorphic registration based on the squared L^2 -distance and normalized cross correlation are reported in the second and third rows, respectively. We report the scores for the individual labels in Figure 9.

	mean	stdev	min	max	median	1st quantile	3rd quantile
initial	0.551	0.041	0.421	0.625	0.555	0.527	0.583
SSD	0.831	0.059	0.697	0.922	0.842	0.785	0.884
NCC	0.835	0.060	0.699	0.923	0.844	0.784	0.889

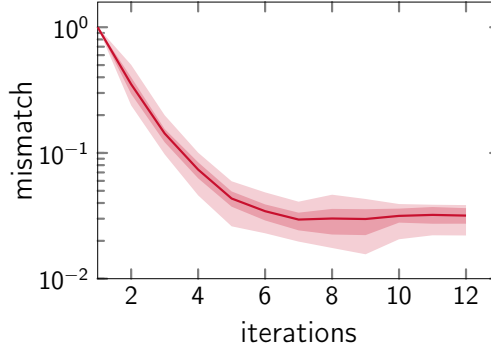


FIGURE 10. Convergence behavior. We plot the relative reduction of the mismatch versus the number of iterations. The plot is generated for 240 registrations. We execute the algorithm for a regularization parameter value of $\alpha = 1.773\text{e-}3$. We solve the problem without using any continuation scheme. The average runtime is 3.615. The solid line represents the mean convergence for the data mismatch. We also show the envelopes for the 25th to 75th quantile and the 5th to the 95th quantile for the values of the mismatch. The average number of iterations is 10.554. We show the trend until iteration 12.

TABLE 4. Workload for estimating the regularization parameter α . We consider a squared L^2 -distance measure. All reported numbers are computed across all 240 registration. We report the number of outer iterations, the number of Hessian matvecs, the number of PDE solves, the relative mismatch after registration, the relative change of the norm of the gradient, and the runtime (in seconds). These numbers are for solving the inverse problem multiple times; we search for an optimal regularization parameter using a binary search.

	mean	stdev	min	max	median	1st QT	3rd QT
iterations	10.554	1.246	9	15	10	10	11
matvecs	24.525	5.659	14	47	24	21	28
PDE solves	82.158	13.664	58	136	82	74	89
mismatch	4.399e-2	1.037e-2	1.906e-2	8.087e-2	4.292e-2	3.691e-2	5.008e-2
gradient	3.940e-2	6.485e-3	2.135e-2	4.973e-2	4.012e-2	3.623e-2	4.409e-2
runtime	3.839	0.648	2.704	6.399	3.779	3.451	4.159

performing any scale, grid, or parameter continuation. We report the runtime for our parameter continuation scheme for a target regularization parameter $\alpha = 1.773\text{e-}3$ in Table 4.

The most important observations are: (i) We can solve the inverse problem in under 4 seconds (on average, the runtime is 3.839 seconds; standard deviation: 0.648 seconds), with a minimum runtime of under 3 seconds and a maximum runtime of slightly above 6 seconds. (ii) We converge in about 12 iterations to a stable solution of our problem (the mismatch stagnates), where a majority of the runs we have executed converge after only 10.554 (standard deviation: 1.246). (iii) Once we have determined an adequate regularization parameter for a particular application, we can solve

the problem quickly with an accuracy that is equivalent to the more expensive parameter search considered in the section above as judged by the relative reduction of the mismatch.

4. CONCLUSIONS

We have reviewed our past work on scalable algorithms for diffeomorphic image registration. Several issues remain.

Our implementation currently only supports the registration of images acquired with the same imaging modality. Developing an effective solver for other distance measures remains subject to future work. We have worked on several numerical schemes for preconditioning the reduced space Hessian. The spectral preconditioner is extremely efficient to apply but its performance deteriorates as we reduce the regularization parameter. This is true for all other schemes we have implemented to precondition the reduced Hessian. Although they are more effective than the simple spectral preconditioner, developing a scheme that has a rate of convergence that is mesh-independent and at the same time independent of the choice of the regularization (parameter) remains subject to future work.

Our 3D GPU implementation currently only supports stationary velocities. These velocities do not define a proper metric in the Riemannian space of diffeomorphic flows. While we have implemented a MATLAB prototype version of a solver that supports time varying velocities, this implementation has not yet been ported to the C++ implementation of CLAIRE.

Another challenge in diffeomorphic image registration is how to handle data that underwent topological changes (e.g., the emergence of a tumor or tissue being removed due to clinical intervention). One possibility to handle this is to introduce additional biophysical constraints [93, 173–182]. On the downside, this makes the problem much more challenging to solve since we not only invert for a deformation map but also for the parameters of the model. More generic approaches to deal with changes in topology are described in [183–187].

Acknowledgements. This work was in part supported by the National Science Foundation (NSF) through the grants DMS-2012825 and DMS-2145845. Any opinions, findings, and conclusions or recommendations expressed herein are those of the author and do not necessarily reflect the views of the NSF. This work was completed in part with resources provided by the Research Computing Data Core at the University of Houston. The author would like to thank George Biros, Malte Brunn, Amir Gholami, Naveen Himthani, Jae Youn Kim, and Miriam Schulte for their numerous contributions to this work.

APPENDIX

PDE Constraints. Below, we will revisit some of the problem formulations we have considered in our past work. These are extensions to the formulation considered in §2.2. We limit the description of our methodology to the most basic formulation for simplicity. The default formulation implemented in our current GPU version in CLAIRE is different [90, 91].

Non-Stationary Velocities. In [30], we consider stationary and non-stationary velocities. For non-stationary velocities, the reduced gradient in (9) is given by

$$\mathbf{g}(\mathbf{v}) := \alpha \mathcal{L} \mathbf{v} + \lambda \nabla m.$$

(Near-)Incompressible Diffeomorphisms. In [30], we augment the formulation in (7) by introducing the incompressibility constraint $\nabla \cdot \mathbf{v} = 0$. A similar formulation has been considered in [40]. For

the primal-dual optimal variables $(m^*, \mathbf{v}^*, \lambda^*, \rho^*)$, the associated KKT conditions are given by

$$\begin{aligned}
(17a) \quad & \partial_t m^* + \mathbf{v}^* \cdot \nabla m^* = 0 && \text{in } (0, 1] \times \Omega, \\
(17b) \quad & m^* = m_0 && \text{in } \{0\} \times \Omega, \\
(17c) \quad & -\partial_t \lambda^* + \nabla \cdot \lambda^* \mathbf{v}^* = 0 && \text{in } [0, 1) \times \Omega, \\
(17d) \quad & \lambda^* = -(m^* - m_1) && \text{in } \{1\} \times \Omega, \\
(17e) \quad & \nabla \cdot \mathbf{v}^* = 0 && \text{in } \Omega, \\
(17f) \quad & \alpha \mathcal{L} \mathbf{v}^* + \nabla \rho^* + \int_0^1 \lambda^* \nabla m^* dt = \mathbf{0} && \text{in } \Omega.
\end{aligned}$$

We eliminate the incompressibility constraint (17e) and the dual variable ρ from the optimality system stated above to obtain the expression

$$\alpha \mathcal{L} \mathbf{v} + \int_0^1 \lambda \nabla m dt - \nabla \Delta^{-1} \nabla \cdot \int_0^1 \lambda \nabla m dt$$

for the reduced gradient. The remaining PDE operators in (17) for m and λ in the associated KKT system are identical.

In [33], we relaxed the incompressibility constraint by introducing an additional control variable w to obtain $\nabla \cdot \mathbf{v} = w$. This allows us to model near-incompressible deformations. After eliminating the constraint $\nabla \cdot \mathbf{v} = w$ and the associated dual variable ρ from the KKT system, we obtain the reduced gradient

$$\alpha \mathcal{L} \mathbf{v}^* + \int_0^1 \lambda^* \nabla m^* dt - \nabla (\alpha (\beta (-\Delta^{-1} + \text{id}))^{-1} + \text{id})^{-1} \Delta^{-1} \nabla \cdot \int_0^1 \lambda^* \nabla m^* dt,$$

Here, $\beta > 0$ denotes the regularization parameter of the regularizer for the second control variable w . We consider an H^1 -norm. We refer to [33] for additional details. This represents the default model implemented in the hardware-accelerated implementation of CLAIRE [1, 90, 91]. The results reported in this study also consider this formulation. The regularization model for the velocity field is an H^1 -seminorm.

Aside from this, we have also explored a model of incompressible flows that promotes shear [33]. To do so, we introduce a nonlinear regularization model. In particular, we replaced the regularization model for \mathbf{v} by

$$|\mathbf{v}|_{H^1(\Omega)}^{(1+\nu)/2\nu} = \frac{2\nu}{\nu+1} \int_{\Omega} (\mathcal{E}[\mathbf{v}] : \mathcal{E}[\mathbf{v}])^{(1+\nu)/2\nu} d\mathbf{x},$$

where

$$\mathcal{E}[\mathbf{v}] := \frac{1}{2} \left((\nabla_d \mathbf{v}) + (\nabla_d \mathbf{v})^\top \right), \quad \nabla_d \mathbf{v} := \begin{pmatrix} (\nabla v_1)^\top \\ \vdots \\ (\nabla v_d)^\top \end{pmatrix} \in \mathbb{R}^{d,d},$$

denotes the strain tensor, and $\nu > 0$ controls the non-linearity. With this regularization model in conjunction with the incompressibility constraint $\nabla \cdot \mathbf{v} = 0$ we obtain a Stokes-like optimality system with a viscosity that depends on the strain rate. The reduced gradient is given by

$$-\text{div} \left(2 \text{tr}(\mathcal{E}[\mathbf{v}] \mathcal{E}[\mathbf{v}])^{(1-\nu)/2\nu} \mathcal{E}[\mathbf{v}] \right) + \nabla \rho + \int_0^1 \lambda \nabla m dt,$$

where $\text{div}(\mathbf{A}) = (\nabla \cdot \mathbf{a}_1, \dots, \nabla \cdot \mathbf{a}_d) \in \mathbb{R}^d$ for an arbitrary $d \times d$ matrix \mathbf{A} with columns $\mathbf{a}_i \in \mathbb{R}^d$, $i = 1, \dots, d$. In the limit $\nu \rightarrow \infty$ this model behaves like total variation regularization. For $\nu \in (0, 1)$ we obtain a shear thickening and for $\nu > 1$ a shear thinning fluid. Likewise to the linear case, we can eliminate the incompressibility constraint and the associated dual variable ρ from the optimality system. We refer to [33] for additional details.

Optimal Transport. In our past work, we have not only introduced new hard or soft constraints for \mathbf{v} but also considered a different forward model for transporting m . In particular, we use the continuity equation

$$\partial_t m + \nabla \cdot m \mathbf{v} = 0 \quad \text{in } [0, 1] \times \Omega,$$

with initial condition $m = m_0$ in $\{0\} \times \Omega$ to model the transport of the intensities of the template image m_0 . In this model, mass is conserved. This establishes a connection to optimal transport [47, 188, 189]. We refer to [48] for additional details.

Deformation Gradient. In the context of image registration, the determinant of the deformation gradient $\det \nabla \mathbf{y}$ is often used to assess invertibility of \mathbf{y} as well as a measure of local volume change in the context of morphometry and shape analysis. In the framework of continuum mechanics, we can obtain this information from the deformation tensor field $\mathbf{f} : [0, 1] \times \bar{\Omega} \rightarrow \mathbb{R}^{d,d}$, where \mathbf{f} is related to \mathbf{v} by

$$(18) \quad \partial_t \mathbf{f} + (\mathbf{v} \cdot \nabla_d) \mathbf{f} = (\nabla_d \mathbf{v}) \mathbf{f} \quad \text{in } \Omega \times (0, 1], \quad \mathbf{f} = \mathbf{I}_d \quad \text{in } \Omega \times \{0\},$$

with periodic boundary conditions on $\partial\Omega$. Here, $\mathbf{I}_d = \text{diag}(1, \dots, 1) \in \mathbb{R}^{d,d}$. In our implementation we use $\det \mathbf{f}_1$ with $\mathbf{f}_1 := \mathbf{f}(t = 1, \cdot)$ as a surrogate for $\det \nabla \mathbf{y}$,

Normalized Cross Correlation. Aside from using the squared L^2 -distance, we also consider normalized cross correlation as a distance measure. We note that we have not presented results for normalized cross correlation elsewhere. The choice of the similarity measure in general only affects the final condition of the dual variable. The normalized cross correlation distance measure is given by

$$(19) \quad \text{dist}_{\text{NCC}}(m(1), m_1) = 1 - \frac{\langle m(1), m_1 \rangle_{L^2(\Omega)}^2}{\langle m_1, m_1 \rangle_{L^2(\Omega)} \langle m(1), m(1) \rangle_{L^2(\Omega)}},$$

where

$$\langle u, w \rangle_{L^2(\Omega)} = \int_{\Omega} u(\mathbf{x}) w(\mathbf{x}) \, d\mathbf{x}$$

denotes the standard L^2 -inner product on $\Omega \subset \mathbb{R}^d$ for arbitrary functions $u : \bar{\Omega} \rightarrow \mathbb{R}$, $w : \bar{\Omega} \rightarrow \mathbb{R}$. Using this distance, the final condition for the adjoint equation is given by

$$\lambda(1, \mathbf{x}) = -2 \frac{\langle m_1, m(1) \rangle_{L^2(\Omega)}}{\|m(1)\|_{L^2(\Omega)}^2 \|m_R\|_{L^2(\Omega)}^2} \left(\frac{\langle m_1, m(1) \rangle_{L^2(\Omega)}}{\|m(1)\|_{L^2(\Omega)}^2} m(1, \mathbf{x}) - m_1(\mathbf{x}) \right).$$

Similarly, the expression for the final condition of the incremental dual variable $\tilde{\lambda}$ is given by

$$\tilde{\lambda}(1, \mathbf{x}) = \frac{2(q_1 m_1(\mathbf{x}) + q_2 m(1, \mathbf{x}) - q_3 \tilde{m}(1, \mathbf{x}))}{\|m_1\|_{L^2(\Omega)}^2},$$

where

$$\begin{aligned} q_1 &= 2 \frac{\langle m_1, m(1) \rangle_{L^2(\Omega)} \langle m(1), \tilde{m}(1) \rangle_{L^2(\Omega)}}{\|m(1)\|_{L^2(\Omega)}^4} - \frac{\langle m_R, \tilde{m}(1) \rangle_{L^2(\Omega)}}{\|m(1)\|_{L^2(\Omega)}^2}, \\ q_2 &= 4 \frac{\langle m_R, m(1) \rangle_{L^2(\Omega)}^2 \langle m(1), \tilde{m}(1) \rangle_{L^2(\Omega)}}{\|m(1)\|_{L^2(\Omega)}^6} - 2 \frac{\langle m_R, m(1) \rangle_{L^2(\Omega)} \langle m_R, \tilde{m}(1) \rangle_{L^2(\Omega)}}{\|m(1)\|_{L^2(\Omega)}^4}, \\ q_3 &= \frac{\langle m_R, m(1) \rangle_{L^2(\Omega)}^2}{\|m(1)\|_{L^2(\Omega)}^4}. \end{aligned}$$

Newton–Krylov Algorithm. We summarize our Newton–Krylov algorithm here. The outer iterations are given in Algorithm 1. The inner iterations (i.e., the computation of the search direction) is given in Algorithm 2. We describe this algorithm in some detail in §2.5.5.

Algorithm 1 Inexact Newton–Krylov method (outer iterations). We use the relative norm of the reduced gradient with tolerance $\epsilon_{\text{opt}} > 0$ as stopping criterion.

```

1:  $k \leftarrow 0$ 
2: initial guess  $\mathbf{v}^{(k)} \leftarrow \mathbf{0}$ 
3:  $\mathbf{m}^{(k)} \leftarrow$  solve state equation in (1b) forward in time given  $\mathbf{v}^{(k)}$ 
4:  $j^{(k)} \leftarrow$  evaluate objective functional (1a) given  $\mathbf{m}^{(k)}$  and  $\mathbf{v}^{(k)}$ 
5:  $\lambda^{(k)} \leftarrow$  solve adjoint equation (10) backward in time given  $\mathbf{v}^{(k)}$  and  $\mathbf{m}^{(k)}$ 
6:  $\mathbf{g}^{(k)} \leftarrow$  evaluate reduced gradient (9) given  $\mathbf{m}^{(k)}$ ,  $\lambda^{(k)}$  and  $\mathbf{v}^{(k)}$ 
7: while  $\|\mathbf{g}^{(k)}\|_{\infty} > \|\mathbf{g}^{(0)}\|_{\infty} \epsilon_{\text{opt}}$  do
8:    $\tilde{\mathbf{v}}^{(k)} \leftarrow$  solve  $\mathbf{H}^{(k)} \tilde{\mathbf{v}}^{(k)} = -\mathbf{g}^{(k)}$  given  $\mathbf{m}^{(k)}$ ,  $\lambda^{(k)}$ ,  $\mathbf{v}^{(k)}$ , and  $\mathbf{g}^{(k)}$  (see Algorithm 2)
9:    $\gamma^{(k)} \leftarrow$  perform line search on  $\tilde{\mathbf{v}}^{(k)}$  subject to Armijo condition
10:   $\mathbf{v}^{(k+1)} \leftarrow \mathbf{v}^{(k)} + \gamma^{(k)} \tilde{\mathbf{v}}^{(k)}$ 
11:   $\mathbf{m}^{(k+1)} \leftarrow$  solve state equation (1b) forward in time given  $\mathbf{v}^{(k+1)}$ 
12:   $j^{(k+1)} \leftarrow$  evaluate (1a) given  $\mathbf{m}^{(k+1)}$  and  $\mathbf{v}^{(k+1)}$ 
13:   $\lambda^{(k+1)} \leftarrow$  solve adjoint equation (10) backward in time given  $\mathbf{v}^{(k+1)}$  and  $\mathbf{m}^{(k+1)}$ 
14:   $\mathbf{g}^{(k+1)} \leftarrow$  evaluate (9) given  $\mathbf{m}^{(k+1)}$ ,  $\lambda^{(k+1)}$  and  $\mathbf{v}^{(k+1)}$ 
15:   $k \leftarrow k + 1$ 
16: end while

```

Algorithm 2 Newton step (inner iterations). We illustrate the solution of the reduced KKT system (15) using a PCG method at a given outer iteration $k \in \mathbb{N}$. We use a superlinear forcing sequence to compute the tolerance $\eta^{(k)}$ for the PCG method (inexact solve).

```

1: input:  $\mathbf{m}^{(k)}$ ,  $\lambda^{(k)}$ ,  $\mathbf{v}^{(k)}$ ,  $\mathbf{g}^{(k)}$ ,  $\mathbf{g}^{(0)}$ 
2:  $l \leftarrow 0$ 
3: set  $\epsilon_H \leftarrow \min(0.5, \|\mathbf{g}^{(k)}\|_{\infty}^{1/2})$ ,  $\tilde{\mathbf{v}}^{(l)} \leftarrow \mathbf{0}$ ,  $\mathbf{r}^{(l)} \leftarrow -\mathbf{g}^{(k)}$ 
4:  $\mathbf{z}^{(l)} \leftarrow$  apply preconditioner  $\mathbf{M}^{-1}$  to  $\mathbf{r}^{(l)}$ 
5:  $\mathbf{s}^{(l)} \leftarrow \mathbf{z}^{(l)}$ 
6: while  $l < n$  do
7:    $\tilde{\mathbf{m}}^{(l)} \leftarrow$  solve (12) forward in time given  $\mathbf{m}^{(k)}$ ,  $\mathbf{v}^{(k)}$  and  $\tilde{\mathbf{v}}^{(l)}$ 
8:    $\tilde{\lambda}^{(l)} \leftarrow$  solve (13) backward in time given  $\lambda^{(k)}$ ,  $\mathbf{v}^{(k)}$ ,  $\tilde{\mathbf{m}}^{(l)}$  and  $\tilde{\mathbf{v}}^{(l)}$ 
9:    $\tilde{\mathbf{s}}^{(l)} \leftarrow$  apply  $\mathbf{H}^{(l)}$  to  $\mathbf{s}^{(l)}$  given  $\lambda^{(k)}$ ,  $\mathbf{m}^{(k)}$ ,  $\tilde{\mathbf{m}}^{(l)}$  and  $\tilde{\lambda}^{(l)}$  (see (11))
10:   $\kappa^{(l)} \leftarrow \langle \mathbf{r}^{(l)}, \mathbf{z}^{(l)} \rangle / \langle \mathbf{s}^{(l)}, \tilde{\mathbf{s}}^{(l)} \rangle$ 
11:   $\tilde{\mathbf{v}}^{(l+1)} \leftarrow \tilde{\mathbf{v}}^{(l)} + \kappa^{(l)} \mathbf{s}^{(l)}$ 
12:   $\mathbf{r}^{(l+1)} \leftarrow \mathbf{r}^{(l)} - \kappa^{(l)} \tilde{\mathbf{s}}^{(l)}$ 
13:  if  $\|\mathbf{r}^{(l+1)}\|_2 < \epsilon_H$  break
14:   $\mathbf{z}^{(l+1)} \leftarrow$  apply preconditioner  $\mathbf{M}^{-1}$  to  $\mathbf{r}^{(l+1)}$ 
15:   $\mu^{(l)} \leftarrow \langle \mathbf{z}^{(l+1)}, \mathbf{r}^{(l+1)} \rangle / \langle \mathbf{z}^{(l)}, \mathbf{r}^{(l)} \rangle$ 
16:   $\mathbf{s}^{(l+1)} \leftarrow \mathbf{z}^{(l+1)} + \mu^{(l)} \mathbf{s}^{(l)}$ 
17:   $l \leftarrow l + 1$ 
18: end while
19: output:  $\tilde{\mathbf{v}}^{(k)} \leftarrow \tilde{\mathbf{v}}^{(l+1)}$ 

```

Hardware. We execture CLAIRE on the Sabine Cluster of the Research Computing Data Core at the University of Houston. Sabine hosts a total of 5704 CPU cores in 169 compute and 12 GPU nodes. We limit the experiments to our GPU implementation. The associated nodes are equipped with a Intel Xeon E5-2680v4 CPUs (2 sockets with 28 cores) with 256 GB of memory. Each node is also equiped with 8 NVIDIA V100 GPUs with a total of 40,960 cores and 128 GB of memory.

TABLE 5. Average DICE values. We report the mean, min, max, and median value as well as the 1st quantile and the 3rd quantile. These values are computed for each individual label for all 240 registrations. These results are obtained for the squared L^2 -distance measure.

label id	mean	stdev	min	max	median	1st quantile	3rd quantile
1	0.661	0.082	0.445	0.838	0.658	0.613	0.719
2	0.678	0.070	0.510	0.833	0.681	0.629	0.732
3	0.678	0.067	0.500	0.822	0.681	0.628	0.724
4	0.689	0.054	0.548	0.794	0.691	0.650	0.731
5	0.765	0.046	0.626	0.852	0.769	0.733	0.803
6	0.789	0.042	0.624	0.867	0.796	0.759	0.822
7	0.738	0.070	0.498	0.876	0.744	0.707	0.787
8	0.786	0.054	0.602	0.888	0.793	0.756	0.826
9	0.642	0.066	0.496	0.820	0.639	0.596	0.690
10	0.629	0.081	0.423	0.811	0.634	0.565	0.691
11	0.732	0.055	0.577	0.858	0.736	0.691	0.774
12	0.729	0.058	0.588	0.865	0.733	0.688	0.774
13	0.734	0.046	0.611	0.839	0.742	0.702	0.763
14	0.768	0.042	0.637	0.861	0.771	0.744	0.798
15	0.736	0.057	0.510	0.835	0.748	0.706	0.779
16	0.721	0.061	0.475	0.828	0.731	0.687	0.763
17	0.680	0.062	0.520	0.813	0.690	0.634	0.727
18	0.673	0.069	0.480	0.804	0.681	0.625	0.721
19	0.655	0.073	0.467	0.794	0.660	0.607	0.715
20	0.621	0.079	0.412	0.789	0.623	0.566	0.678
21	0.569	0.122	0.272	0.841	0.556	0.488	0.669
22	0.602	0.081	0.359	0.790	0.609	0.558	0.660
23	0.750	0.054	0.571	0.851	0.754	0.715	0.792
24	0.746	0.053	0.596	0.861	0.747	0.703	0.792
25	0.604	0.082	0.375	0.792	0.608	0.549	0.663
26	0.562	0.091	0.271	0.758	0.569	0.501	0.629
27	0.600	0.098	0.324	0.792	0.607	0.540	0.681
28	0.587	0.084	0.370	0.777	0.594	0.524	0.652
29	0.650	0.078	0.440	0.805	0.660	0.592	0.714
30	0.618	0.083	0.412	0.814	0.621	0.564	0.675
31	0.547	0.093	0.220	0.741	0.555	0.485	0.611
32	0.486	0.103	0.224	0.726	0.490	0.415	0.566
33	0.656	0.057	0.507	0.780	0.657	0.620	0.697

Additional Results. We report more detailed results for the registration accuracy of CLAIRE in this section. The statistics for the Dice for the squared L^2 -distance with respect to each individual label is reported in Table 5. The associated results for normalized cross correlation are reported in Table 6. These results are for the parameter search for 240 registration (all-to-all) of the NIREP dataset.

REFERENCES

- [1] A. Mang et al. “CLAIRE: A distributed-memory solver for constrained large deformation diffeomorphic image registration”. In: *SIAM Journal on Scientific Computing* 41.5 (2019), pp. C548–C584 (cit. on pp. 1, 4, 5, 8, 10–18, 25).

TABLE 6. Average DICE values. We report the mean, min, max, and median value as well as the 1st quantile and the 3rd quantile. These values are computed for each individual label for all 240 registrations. The results are obtained for the normalized cross correlation distance measure.

label id	mean	stdev	min	max	median	1st quantile	3rd quantile
1	0.665	0.082	0.427	0.835	0.660	0.614	0.724
2	0.681	0.070	0.491	0.822	0.683	0.632	0.735
3	0.683	0.066	0.514	0.834	0.690	0.639	0.727
4	0.694	0.052	0.562	0.808	0.697	0.655	0.734
5	0.772	0.043	0.628	0.850	0.780	0.747	0.804
6	0.790	0.042	0.620	0.867	0.798	0.759	0.824
7	0.743	0.067	0.543	0.877	0.758	0.707	0.791
8	0.787	0.053	0.605	0.888	0.796	0.756	0.827
9	0.648	0.066	0.499	0.816	0.649	0.597	0.696
10	0.632	0.080	0.425	0.810	0.633	0.573	0.688
11	0.736	0.055	0.588	0.858	0.743	0.696	0.778
12	0.731	0.059	0.587	0.864	0.734	0.690	0.773
13	0.740	0.044	0.615	0.847	0.743	0.710	0.771
14	0.773	0.042	0.639	0.860	0.775	0.749	0.803
15	0.740	0.059	0.504	0.842	0.751	0.703	0.786
16	0.725	0.063	0.469	0.837	0.739	0.689	0.772
17	0.686	0.063	0.522	0.822	0.696	0.642	0.735
18	0.678	0.068	0.493	0.806	0.681	0.632	0.727
19	0.660	0.073	0.471	0.799	0.665	0.611	0.722
20	0.625	0.080	0.414	0.795	0.631	0.572	0.684
21	0.576	0.120	0.271	0.841	0.574	0.495	0.672
22	0.604	0.083	0.328	0.795	0.613	0.553	0.667
23	0.754	0.056	0.566	0.853	0.758	0.716	0.796
24	0.749	0.054	0.597	0.861	0.752	0.707	0.791
25	0.609	0.081	0.377	0.794	0.617	0.550	0.661
26	0.565	0.092	0.263	0.768	0.570	0.501	0.636
27	0.603	0.095	0.331	0.786	0.610	0.550	0.682
28	0.589	0.083	0.364	0.778	0.593	0.530	0.646
29	0.653	0.080	0.448	0.817	0.667	0.589	0.719
30	0.618	0.083	0.418	0.815	0.617	0.563	0.678
31	0.553	0.093	0.219	0.731	0.566	0.488	0.622
32	0.488	0.102	0.224	0.740	0.492	0.415	0.570
33	0.662	0.053	0.538	0.780	0.662	0.625	0.699

- [2] A. Mang. *CLAIRE: Constrained Large Deformation Diffeomorphic Image Registration*. <https://andreasmang.github.io/claire>. 2019 (cit. on pp. 1, 4, 5).
- [3] M. Brunn et al. “CLAIRE: Constrained large deformation diffeomorphic image registration on parallel architectures”. In: *The Journal of Open Source Software* 6.61 (2021), p. 3038 (cit. on pp. 1, 4, 5).
- [4] B. Fischer and J. Modersitzki. “Ill-posed medicine – an introduction to image registration”. In: *Inverse Problems* 24.3 (2008), pp. 1–16 (cit. on pp. 1, 3).
- [5] J. Modersitzki. *Numerical methods for image registration*. New York: Oxford University Press, 2004 (cit. on pp. 1, 3).
- [6] J. Modersitzki. *FAIR: Flexible algorithms for image registration*. Philadelphia, Pennsylvania, US: SIAM, 2009 (cit. on pp. 1, 3).
- [7] L. Younes. *Shapes and diffeomorphisms*. 2nd ed. Vol. 171. Springer Verlag Berlin Heidelberg, 2019 (cit. on pp. 1, 3, 5, 6).

- [8] M. F. Beg et al. “Computing large deformation metric mappings via geodesic flows of diffeomorphisms”. In: *International Journal of Computer Vision* 61.2 (2005), pp. 139–157 (cit. on pp. 1, 3–5, 7, 8, 10).
- [9] A. Trouvé. *A infinite dimensional group approach for physics based models in pattern recognition*. Tech. rep. Laboratoire d’Analyse Numerique CNRS URA, Université Paris, 1995 (cit. on pp. 1, 3, 5, 6).
- [10] A. Trouvé. “Diffeomorphism groups and pattern matching in image analysis”. In: *International Journal of Computer Vision* 28.3 (1998), pp. 213–221 (cit. on pp. 1, 3, 5).
- [11] P. Dupuis, U. Gernander, and M. I. Miller. “Variational problems on flows of diffeomorphisms for image matching”. In: *Quarterly of Applied Mathematics* 56.3 (1998), pp. 587–600 (cit. on pp. 1, 3, 6, 7).
- [12] A. Borzi and V. Schulz. *Computational optimization of systems governed by partial differential equations*. Philadelphia, Pennsylvania, US: SIAM, 2012 (cit. on pp. 1, 3).
- [13] M. Hinze et al. *Optimization with PDE constraints*. Berlin, DE: Springer, 2009 (cit. on pp. 1, 3).
- [14] H. Antil et al. *Frontiers in PDE-constrained optimization*. Vol. 163. Springer, 2018 (cit. on pp. 1, 3).
- [15] A. Mang et al. “PDE-constrained optimization in medical image analysis”. In: *Optimization and Engineering* 19.3 (2018), pp. 765–812 (cit. on pp. 1, 4, 15).
- [16] L. T. Biegler et al. *Large-scale PDE-constrained optimization*. Springer, 2003 (cit. on pp. 3, 4).
- [17] M. D. Gunzburger. *Perspectives in flow control and optimization*. Philadelphia, Pennsylvania, US: SIAM, 2003 (cit. on pp. 3, 8).
- [18] J. L. Lions. *Optimal control of systems governed by partial differential equations*. Springer, 1971 (cit. on p. 3).
- [19] J. V. Hajnal, D. L. G. Hill, and D. J. Hawkes, eds. *Medical Image Registration*. Boca Raton, Florida, US: CRC Press, 2001 (cit. on p. 3).
- [20] A. Sotiras, C. Davatzikos, and N. Paragios. “Deformable medical image registration: A survey”. In: *Medical Imaging, IEEE Transactions on* 32.7 (2013), pp. 1153–1190 (cit. on p. 3).
- [21] M. Burger, J. Modersitzki, and L. Ruthotto. “A hyperelastic regularization energy for image registration”. In: *SIAM Journal on Scientific Computing* 35.1 (2013), B132–B148 (cit. on p. 3).
- [22] E. Haber and J. Modersitzki. “Image registration with guaranteed displacement regularity”. In: *International Journal of Computer Vision* 71.3 (2007), pp. 361–372 (cit. on p. 3).
- [23] T. Rohlfing et al. “Volume-preserving nonrigid registration of MR breast images using free-form deformation with an incompressibility constraint”. In: *Medical Imaging, IEEE Transactions on* 22 (2003), pp. 730–741 (cit. on p. 3).
- [24] M. Sdika. “A fast nonrigid image registration with constraints on the Jacobian using large scale constrained optimization”. In: *Medical Imaging, IEEE Transactions on* 27.2 (2008), pp. 271–281 (cit. on p. 3).
- [25] L. Younes, B. Gris, and A. Trouvé. “Sub-Riemannian methods in shape analysis”. In: *Handbook of Variational Methods for Nonlinear Geometric Data* (2020), pp. 463–495 (cit. on pp. 3, 6).
- [26] M. I. Miller and L. Younes. “Group actions, homeomorphism, and matching: A general framework”. In: *International Journal of Computer Vision* 41.1/2 (2001), pp. 61–81 (cit. on p. 3).
- [27] T. Vercauteren et al. “Diffeomorphic demons: Efficient non-parametric image registration”. In: *NeuroImage* 45.1 (2009), S61–S72 (cit. on pp. 3, 4, 8).

- [28] G. E. Christensen, R. D. Rabbitt, and M. I. Miller. “Deformable templates using large deformation kinematics”. In: *IEEE Transactions on Image Processing* 5.10 (1996), pp. 1435–1447 (cit. on p. 3).
- [29] L. Younes. “Jacobi fields in groups of diffeomorphisms and applications”. In: *Quarterly of Applied Mathematics* 650.1 (2007), pp. 113–134 (cit. on p. 3).
- [30] A. Mang and G. Biros. “An inexact Newton–Krylov algorithm for constrained diffeomorphic image registration”. In: *SIAM Journal on Imaging Sciences* 8.2 (2015), pp. 1030–1069 (cit. on pp. 3–5, 7–11, 13–15, 17, 24).
- [31] G. L. Hart, C. Zach, and M. Niethammer. “An optimal control approach for deformable registration”. In: *Proc IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 9–16 (cit. on pp. 3, 4, 8).
- [32] A. Borzi, K. Ito, and K. Kunisch. “Optimal control formulation for determining optical flow”. In: *SIAM Journal on Scientific Computing* 24.3 (2002), pp. 818–847 (cit. on pp. 3, 4, 8).
- [33] A. Mang and G. Biros. “Constrained H^1 -regularization schemes for diffeomorphic image registration”. In: *SIAM Journal on Imaging Sciences* 9.3 (2016), pp. 1154–1194 (cit. on pp. 3, 4, 8, 13, 15, 18, 25).
- [34] K. Chen and D. A. Lorenz. “Image sequence interpolation using optimal control”. In: *Journal of Mathematical Imaging and Vision* 41 (2011), pp. 222–238 (cit. on pp. 3, 4).
- [35] J. Hinkle et al. “4D MAP image reconstruction incorporating organ motion”. In: *Proc Information Processing in Medical Imaging*. LNCS 5636. 2009, pp. 676–687 (cit. on p. 3).
- [36] T. Mansi et al. “iLogDemos: A demons-based registration algorithm for tracking incompressible elastic biological tissues”. In: *International Journal of Computer Vision* 92.1 (2011), pp. 92–111 (cit. on p. 3).
- [37] P. Ruhnau and C. Schnörr. “Optical Stokes flow estimation: An imaging-based control approach”. In: *Experiments in Fluids* 42 (2007), pp. 61–78 (cit. on p. 3).
- [38] K. A. Saddi, C. Ched’hotel, and F. Cheriet. “Large deformation registration of contrast-enhanced images with volume-preserving constraint”. In: *Proc SPIE Medical Imaging*. Vol. 6512. 2008, pp. 651203-1–651203-10 (cit. on p. 3).
- [39] A. Borzi, K. Ito, and K. Kunisch. “An optimal control approach to optical flow computation”. In: *International Journal for Numerical Methods in Fluids* 40.1–2 (2002), pp. 231–240 (cit. on pp. 3, 8).
- [40] K. Chen and D. A. Lorenz. “Image sequence interpolation based on optical flow, segmentation and optimal control”. In: *Image Processing, IEEE Transactions on* 21.3 (2012), pp. 1020–1030 (cit. on pp. 3, 8, 24).
- [41] E. Lee and M. Gunzburger. “An optimal control formulation of an image registration problem”. In: *Journal of Mathematical Imaging and Vision* 36.1 (2010), pp. 69–80 (cit. on p. 3).
- [42] E. Lee and M. Gunzburger. “Analysis of finite element discretization of an optimal control formulation of the image registration problem”. In: *SIAM Journal on Numerical Analysis* 49.4 (2011), pp. 1321–1349 (cit. on p. 3).
- [43] F.-X. Vialard et al. “Diffeomorphic 3D image registration via geodesic shooting using an efficient adjoint calculation”. In: *International Journal of Computer Vision* 97 (2012), pp. 229–241 (cit. on pp. 3, 4).
- [44] R. Herzog, J. W. Pearson, and M. Stoll. “Fast iterative solvers for an optimal transport problem”. In: *Advances in Computational Mathematics* 45 (2019), pp. 495–517 (cit. on pp. 3, 4).
- [45] M. Benzi, E. Haber, and L. Taralli. “A preconditioning technique for a class of PDE-constrained optimization problems”. In: *Advances in Computational Mathematics* 35.2-4 (2011), pp. 149–173 (cit. on pp. 3, 4).

- [46] E. Haber and R. Horesh. “A multilevel method for the solution of time dependent optimal transport”. In: *Numerical Mathematics: Theory, Methods and Applications* 8.1 (2015), pp. 97–111 (cit. on p. 3).
- [47] T. ur Rehman et al. “3D nonrigid registration via optimal mass transport on the GPU”. In: *Medical Image Analysis* 13.6 (2009), pp. 931–940 (cit. on pp. 3, 4, 26).
- [48] A. Mang and L. Ruthotto. “A Lagrangian Gauss–Newton–Krylov solver for mass- and intensity-preserving diffeomorphic image registration”. In: *SIAM Journal on Scientific Computing* 39.5 (2017), B860–B885 (cit. on pp. 3–5, 9, 11, 14–16, 26).
- [49] B. K. P. Horn and B. G. Shunck. “Determining optical flow”. In: *Artificial Intelligence* 17.1–3 (1981), pp. 185–203 (cit. on p. 3).
- [50] E. M. Kalmoun, L. Garrido, and V. Caselles. “Line search multilevel optimization as computational methods for dense optical flow”. In: *SIAM Journal on Imaging Sciences* 4.2 (2011), pp. 695–722 (cit. on p. 3).
- [51] R. Andreev, O. Scherzer, and W. Zulehner. “Simultaneous optical flow and source estimation: Space–time discretization and preconditioning”. In: *Applied Numerical Mathematics* 96 (2015), pp. 72–81 (cit. on p. 3).
- [52] V. Barbu and G. Marinoschi. “An optimal control approach to the optical flow problem”. In: *Systems & Control Letters* 87 (2016), pp. 1–9 (cit. on p. 3).
- [53] T. Vercauteren et al. “Symmetric log-domain diffeomorphic registration: A demons-based approach”. In: *Proc Medical Image Computing and Computer-Assisted Intervention*. Vol. LNCS 5241. 5241. 2008, pp. 754–761 (cit. on p. 3).
- [54] B. B. Avants et al. “A reproducible evaluation of ANTs similarity metric performance in brain image registration”. In: *NeuroImage* 54 (2011), pp. 2033–2044 (cit. on p. 3).
- [55] B. B. Avants et al. “Symmetric diffeomorphic image registration with cross-correlation: Evaluating automated labeling of elderly and neurodegenerative brain”. In: *Medical Image Analysis* 12.1 (2008), pp. 26–41 (cit. on p. 3).
- [56] A. Bône et al. “Deformetrica 4: An open-source software for statistical shape analysis”. In: *International Workshop on Shape in Medical Imaging*. Springer. 2018, pp. 3–13 (cit. on pp. 3, 4).
- [57] J. Fishbaugh et al. “Geodesic shape regression with multiple geometries and sparse parameters”. In: *Medical Image Analysis* 39 (2017), pp. 1–17 (cit. on p. 3).
- [58] J. Ashburner. “A fast diffeomorphic image registration algorithm”. In: *NeuroImage* 38.1 (2007), pp. 95–113 (cit. on pp. 3, 8).
- [59] Y. Cao et al. “Large deformation diffeomorphic metric mapping of vector fields”. In: *Medical Imaging, IEEE Transactions on* 24.9 (2005), pp. 1216–1230 (cit. on p. 4).
- [60] D.-N. Hsieh et al. “Mechanistic modeling of longitudinal shape changes: Equations of motion and inverse problems”. In: *SIAM Journal on Applied Dynamical Systems* 21.1 (2022), pp. 80–101 (cit. on p. 4).
- [61] T. Polzin et al. “Memory efficient LDDMM for lung CT”. In: *Proc Medical Image Computing and Computer-Assisted Intervention*. Vol. LNCS 9902. 2016, pp. 28–36 (cit. on p. 4).
- [62] T. Polzin et al. “A discretize–optimize approach for LDDMM registration”. In: *Riemannian Geometric Statistics in Medical Image Analysis*. Elsevier, 2020, pp. 479–532 (cit. on pp. 4, 7, 9).
- [63] M. Niethammer, G. L. Hart, and C. Zach. “An optimal control approach for the registration of image time-series”. In: *Proceedings of the 48th IEEE Conference on Decision and Control*. IEEE. 2009, pp. 2427–2434 (cit. on p. 4).
- [64] S. Arguillere, M. I. Miller, and L. Younes. “Diffeomorphic surface registration with atrophy constraints”. In: *SIAM Journal on Imaging Sciences* 9.3 (2016), pp. 975–1003 (cit. on pp. 4, 7).

- [65] M. Zhang and P. T. Fletcher. “Finite-dimensional Lie algebras for fast diffeomorphic image registration”. In: *Proc Information Processing in Medical Imaging*. Vol. 24. 2015, pp. 249–259 (cit. on p. 4).
- [66] J. Ashburner and K. J. Friston. “Diffeomorphic registration using geodesic shooting and Gauss-Newton optimisation”. In: *NeuroImage* 55.3 (2011), pp. 954–967 (cit. on p. 4).
- [67] M. I. Miller, A. Trounev, and L. Younes. “Geodesic shooting for computational anatomy”. In: *Journal of Mathematical Imaging and Vision* 24 (2006), pp. 209–228 (cit. on p. 4).
- [68] R. Azencott et al. “Diffeomorphic matching and dynamic deformable surfaces in 3D medical imaging”. In: *Computational Methods in Applied Mathematics* 10.3 (2010), pp. 235–274 (cit. on pp. 4, 7, 17).
- [69] P. Zhang et al. “Diffeomorphic shape matching by operator splitting in 3D cardiology imaging”. In: *Journal of Optimization Theory and Applications* 188 (2021), pp. 143–168 (cit. on pp. 4, 7).
- [70] A. Mang, J. He, and R. Azencott. “An operator-splitting approach for variational optimal control formulations for diffeomorphic shape matching”. In: *Journal of Computational Physics* (2023) (cit. on pp. 4, 7).
- [71] A. Mang, A. Gholami, and G. Biros. “Distributed-memory large-deformation diffeomorphic 3D image registration”. In: *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*. 2016, pp. 842–853 (cit. on pp. 4, 10–12, 15).
- [72] A. Mang and G. Biros. “A semi-Lagrangian two-level preconditioned Newton–Krylov solver for constrained diffeomorphic image registration”. In: *SIAM Journal on Scientific Computing* 39.6 (2017), B1064–B1101 (cit. on pp. 4, 10, 11).
- [73] M. Hernandez. “Gauss-Newton inspired preconditioned optimization in large deformation diffeomorphic metric mapping”. In: *Physics in Medicine and Biology* 59.20 (2014), pp. 6085–6115 (cit. on p. 4).
- [74] V. Simoncini. “Reduced order solution of structured linear systems arising in certain PDE-constrained optimization problems”. In: *Computational Optimization and Applications* 53.2 (2012), pp. 591–617 (cit. on p. 4).
- [75] A. Thorley et al. “Nesterov accelerated ADMM for fast diffeomorphic image registration”. In: *Medical Image Computing and Computer Assisted Intervention*. 2021, pp. 150–160 (cit. on p. 4).
- [76] Y. T. Lee, K. C. Lam, and L. M. Lui. “Landmark-matching transformation with large deformation via n -dimensional quasi-conformal maps”. In: *Journal of Scientific Computing* 67 (2016), pp. 926–954 (cit. on p. 4).
- [77] H.-W. Hsieh and N. Charon. “Diffeomorphic registration with density changes for the analysis of imbalanced shapes”. In: *International Conference on Information Processing in Medical Imaging*. Springer. 2021, pp. 31–42 (cit. on p. 4).
- [78] A. François, P. Gori, and J. Glaunès. “Metamorphic image registration using a semi-Lagrangian scheme”. In: *International Conference on Geometric Science of Information*. Springer. 2021, pp. 781–788 (cit. on pp. 4, 10).
- [79] E. Hartman et al. “Elastic shape analysis of surfaces with second-order Sobolev metrics: A comprehensive numerical framework”. In: *International Journal of Computer Vision* (2023), pp. 1–27 (cit. on pp. 4, 7).
- [80] A. Bône et al. “Learning the spatiotemporal variability in longitudinal shape data sets”. In: *International Journal of Computer Vision* 128.12 (2020), pp. 2873–2896 (cit. on p. 4).
- [81] Z. Shen et al. “Accurate point cloud registration with robust optimal transport”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 5373–5389 (cit. on p. 4).
- [82] L. Tian et al. In: (2023), pp. 18084–18094 (cit. on p. 4).

- [83] B. B. Amor, S. Arguillère, and L. Shao. “ResNet-LDDMM: Advancing the LDDMM framework using deep residual networks”. In: *arXiv preprint arXiv:2102.07951* (2021) (cit. on p. 4).
- [84] J. Krebs et al. “Learning a probabilistic model for diffeomorphic registration”. In: *IEEE Transactions on Medical Imaging* 38.9 (2019), pp. 2165–2176 (cit. on p. 4).
- [85] S. Sun et al. “Topology-preserving shape reconstruction and registration via neural diffeomorphic flow”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 20845–20855 (cit. on p. 4).
- [86] X. Yang et al. “Quicksilver: Fast predictive image registration—A deep learning approach”. In: *NeuroImage* 158 (2017), pp. 378–396 (cit. on p. 4).
- [87] N. Wu and M. Zhang. “NeurEPDiff: Neural Operators to Predict Geodesics in Deformation Spaces”. In: *International Conference on Information Processing in Medical Imaging*. Springer. 2023, pp. 588–600 (cit. on p. 4).
- [88] S. Bharati et al. “Deep learning for medical image registration: A comprehensive review”. In: *arXiv preprint arXiv:2204.11341* (2022) (cit. on p. 4).
- [89] Y. Wu et al. “NODEO: A neural ordinary differential equation based optimization framework for deformable image registration”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 20804–20813 (cit. on p. 4).
- [90] M. Brunn et al. “Multi-node multi-GPU diffeomorphic image registration for large-scale imaging problems”. In: *Proc ACM/IEEE Conference on Supercomputing*. 2020, pp. 523–539 (cit. on pp. 4, 8, 10–13, 16, 18, 24, 25).
- [91] M. Brunn et al. “Fast GPU 3D diffeomorphic image registration”. In: *Journal of Parallel and Distributed Computing* 149 (2021), pp. 149–162 (cit. on pp. 4, 8, 10–13, 24, 25).
- [92] M. Benzi, G. H. Golub, and J. Liesen. “Numerical solution of saddle point problems”. In: *Acta Numerica* 14 (2005), pp. 1–137 (cit. on p. 4).
- [93] A. Gholami et al. “A framework for scalable biophysics-based image analysis”. In: *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*. 19. 2017, 19:1–19:13 (cit. on pp. 4, 10, 12, 13, 18, 24).
- [94] V. Akcelik, G. Biros, and O. Ghattas. “Parallel multiscale Gauss-Newton-Krylov methods for inverse wave propagation”. In: *Proc ACM/IEEE Conference on Supercomputing*. 2002, pp. 1–15 (cit. on p. 4).
- [95] V. Akcelik et al. “Parallel algorithms for PDE constrained optimization”. In: ed. by M. A. Heroux, P. Raghavan, and H. D. Simon. Vol. 20. *Parallel Processing for Scientific Computing*. Philadelphia, Pennsylvania, US: SIAM, 2006. Chap. 16, pp. 291–322 (cit. on p. 4).
- [96] G. Biros and O. Ghattas. “Parallel Newton-Krylov methods for PDE-constrained optimization”. In: *Proc ACM/IEEE Conference on Supercomputing*. 1999, pp. 28–40 (cit. on p. 4).
- [97] G. Biros and O. Ghattas. “Parallel Lagrange-Newton-Krylov-Schur methods for PDE-constrained optimization—Part I: The Krylov-Schur solver”. In: *SIAM Journal on Scientific Computing* 27.2 (2005), pp. 687–713 (cit. on p. 4).
- [98] G. Biros and O. Ghattas. “Parallel Lagrange-Newton-Krylov-Schur methods for PDE-constrained optimization—Part II: The Lagrange-Newton solver and its application to optimal control of steady viscous flows”. In: *SIAM Journal on Scientific Computing* 27.2 (2005), pp. 714–739 (cit. on p. 4).
- [99] L. T. Biegler et al. *Real-time PDE-constrained optimization*. SIAM, 2007 (cit. on p. 4).
- [100] O. Shenk et al. “Parallel scalable PDE-constrained optimization: Antenna identification in hyperthermia cancer treatment planning”. In: *Computer Science—Research and Development* 23.3–4 (2009), pp. 177–183 (cit. on p. 4).
- [101] A. Eklund et al. “Medical image processing on the GPU—past, present and future”. In: *Medical Image Analysis* 17.8 (2013), pp. 1073–1094 (cit. on p. 4).

- [102] O. Fluck et al. “A survey of medical image registration on graphics hardware”. In: *Computer Methods and Programs in Biomedicine* 104.3 (2011), e45–e57 (cit. on p. 4).
- [103] J. Shackleford, N. Kandasamy, and G. Sharp. *High performance deformable image registration algorithms for manycore processors*. Waltham, Massachusetts, US: Morgan Kaufmann, 2013 (cit. on p. 4).
- [104] R. Shams et al. “A survey of medical image registration on multicore and the GPU”. In: *Signal Processing Magazine, IEEE* 27.2 (2010), pp. 50–60 (cit. on p. 4).
- [105] J. Shackleford, N. Kandasamy, and G. Sharp. “On developing B-spline registration algorithms for multi-core processors”. In: *Physics in Medicine and Biology* 55.21 (2010), pp. 6329–6351 (cit. on p. 4).
- [106] M. Modat et al. “Fast free-form deformation using graphics processing units”. In: *Computer Methods and Programs in Biomedicine* 98.3 (2010), pp. 278–284 (cit. on p. 4).
- [107] D. P. Shamonin et al. “Fast parallel image registration on CPU and GPU for diagnostic classification of Alzheimer’s disease”. In: *Frontiers in Neuroinformatics* 7.50 (2014), pp. 1–15 (cit. on p. 4).
- [108] L. K. Ha et al. “Fast parallel unbiased diffeomorphic atlas construction on multi-graphics processing units”. In: *Proc Eurographics Conference on Parallel Graphics and Visualization*. 2009, pp. 41–48 (cit. on p. 4).
- [109] L. Ha et al. “Multiscale unbiased diffeomorphic atlas construction on multi-GPUs”. In: *CPU Computing Gems Emerald Edition*. Elsevier Inc, 2011. Chap. 48, pp. 771–791 (cit. on p. 4).
- [110] S. Sommer. “Accelerating multi-scale flows for LDDKBM diffeomorphic registration”. In: *Proc IEEE International Conference on Computer Visions Workshops*. 2011, pp. 499–505 (cit. on p. 4).
- [111] P. Valero-Lara. “Multi-GPU acceleration of DARTEL (early detection of Alzheimer)”. In: *Proc IEEE International Conference on Cluster Computing*. 2014, pp. 346–354 (cit. on p. 4).
- [112] N. Himthani et al. “CLAIRE: Parallelized diffeomorphic image registration for large-scale biomedical imaging applications”. In: *Journal of Imaging* 8.9 (2022), p. 251 (cit. on pp. 4, 8, 17).
- [113] T. Munson et al. *TAO 3.7 users manual*. Argonne National Laboratory, Mathematics and Computer Science Division. 2017 (cit. on p. 4).
- [114] R. T. Mills et al. “Toward performance-portable PETSc for GPU-based exascale systems”. In: *Parallel Computing* 108 (2021), p. 102831. ISSN: 0167-8191 (cit. on p. 4).
- [115] S. Balay et al. *PETSc Web page*. 2023. URL: <https://petsc.org/> (cit. on pp. 4, 14).
- [116] S. Balay et al. *PETSc/TAO Users Manual*. Tech. rep. ANL-21/39 - Revision 3.20. Argonne National Laboratory, 2023 (cit. on pp. 4, 14).
- [117] J. Glaunès, A. Trouvé, and L. Younes. “Diffeomorphic matching of distributions: A new approach for unlabelled point-sets and sub-manifolds matching”. In: *Proc IEEE Conference on Computer Vision and Pattern Recognition*. Vol. 2. 2004, pp. 712–718 (cit. on p. 5).
- [118] J. Glaunès et al. “Large deformation diffeomorphic metric curve mapping”. In: *International Journal of Computer Vision* 80.3 (2008), pp. 317–336 (cit. on pp. 5, 6).
- [119] V. I. Arnold. “Sur la géométrie différentielle des groupes de Lie de dimension infinie et ses applications a l’hydrodynamique des fluides parfaits”. In: *Annales de l’Institut Fourier* 16 (1966), pp. 319–361 (cit. on p. 5).
- [120] V. I. Arnold. *Les méthodes mathématiques de la mécanique classique*. MIR, Moscow, 1976 (cit. on p. 5).
- [121] D. G. Ebin and J. Marsden. “Groups of diffeomorphisms and the motion of an incompressible fluid”. In: *Annals of Mathematics* 92.1 (1970), pp. 102–163 (cit. on p. 5).
- [122] W. P. Ziemer. *Weakly differentiable functions: Sobolev spaces and functions of bounded variation*. Vol. 120. Springer Science & Business Media, 1989 (cit. on p. 6).

- [123] M. I. Miller, A. Trouvé, and L. Younes. “On the metrics and Euler–Lagrange equations of computational anatomy”. In: *Annual Review of Biomedical Engineering* 4.1 (2002), pp. 375–405 (cit. on pp. 6, 7).
- [124] M. Bauer, M. Bruveris, and P. W. Michor. “Constructing reparametrization invariant metrics on spaces of plane curves”. In: *Differential Geometry and its applications* 34 (2014), pp. 139–165 (cit. on p. 6).
- [125] U. Grenander and M. I. Miller. “Computational anatomy: An emerging discipline”. In: *Quarterly of Applied Mathematics* 56.4 (1998), pp. 617–694 (cit. on p. 7).
- [126] M. I. Miller. “Computational anatomy: Shape, growth and atrophy comparison via diffeomorphisms”. In: *NeuroImage* 23.1 (2004), S19–S33 (cit. on p. 7).
- [127] L. Younes, F. Arrate, and M. I. Miller. “Evolutions equations in computational anatomy”. In: *NeuroImage* 45 (2009), S40–S50 (cit. on p. 7).
- [128] M. I. Miller, A. Trouvé, and L. Younes. “Hamiltonian systems and optimal control in computational anatomy: 100 years since D’Arcy Thompson”. In: *Annual Review of Biomedical Engineering* 17.447–509 (2015) (cit. on p. 7).
- [129] J. Glaunès, M. Vailland, and M. I. Miller. “Landmark matching via large deformation diffeomorphisms on the sphere”. In: *Journal of Mathematical Imaging and Vision* 20 (2004), pp. 179–200 (cit. on p. 7).
- [130] S. Joshi and M. I. Miller. “Landmark matching via large deformation diffeomorphisms”. In: *IEEE Transactions on Image Processing* 9.8 (2000), pp. 1357–1370 (cit. on p. 7).
- [131] S. Durrleman. “Statistical models of currents for measuring the variability of anatomical curves, surfaces and their evolution”. PhD thesis. Université Nice-Sophia Antipolis, France, 2010 (cit. on p. 7).
- [132] S. Durrleman et al. “A forward model to build unbiased atlases from curves and surfaces”. In: *Proc Medical Image Computing and Computer-Assisted Intervention*. 2008, pp. 68–79 (cit. on p. 7).
- [133] J. A. Glaunès and S. Joshi. “Template estimation from unlabeled point set data and surfaces for computational anatomy”. In: *Proc International Workshop on the Mathematical Foundations of Computational Anatomy*. 2006, pp. 29–39 (cit. on p. 7).
- [134] S. Kurtek et al. “Elastic geodesic paths in shape space of parameterized surfaces”. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 34.9 (2012), pp. 1717–1730 (cit. on p. 7).
- [135] Y. Cao et al. “Diffeomorphic matching of diffusion tensor images”. In: *2006 Conference on Computer Vision and Pattern Recognition Workshop*. IEEE. 2006, pp. 67–67 (cit. on p. 7).
- [136] V. Arsigny et al. “A Log-Euclidean framework for statistics on diffeomorphisms”. In: *Proc Medical Image Computing and Computer-Assisted Intervention*. Vol. LNCS 4190. 2006, pp. 924–931 (cit. on p. 8).
- [137] M. Hernandez, M. N. Bossa, and S. Olmos. “Registration of anatomical images using paths of diffeomorphisms parameterized with stationary vector field flows”. In: *International Journal of Computer Vision* 85.3 (2009), pp. 291–306 (cit. on p. 8).
- [138] M. Lorenzi and X. Pennec. “Geodesics, parallel transport and one-parameter subgroups for diffeomorphic image registration”. In: *International Journal of Computer Vision* 105.2 (2013), pp. 111–127 (cit. on p. 8).
- [139] M. Lorenzi et al. “LCC-Demons: a robust and accurate symmetric diffeomorphic registration algorithm”. In: *NeuroImage* 81 (2013), pp. 470–483 (cit. on p. 8).
- [140] A. Staniforth and J. Côté. “Semi-Lagrangian integration schemes for atmospheric models—A review”. In: *Monthly Weather Review* 119.9 (1991), pp. 2206–2223 (cit. on p. 10).
- [141] C. Sigg and M. Hadwiger. “Fast third-order texture filtering”. In: vol. 2. *GPU Gems*. 2005, pp. 313–329 (cit. on p. 13).

- [142] D. Ruijters, B. M. ter Haar Romeny, and P. Suetens. “Efficient GPU-based texture interpolation using uniform B-splines”. In: *Journal of Graphics Tools* 13.4 (2008), pp. 61–69 (cit. on p. 13).
- [143] D. Ruijters and P. Thévenaz. “GPU prefilter for accurate cubic B-spline interpolation”. In: *The Computer Journal* 55.1 (2012), pp. 15–20 (cit. on p. 13).
- [144] F. Champagnat and Y. Le Sant. “Efficient cubic B-spline image interpolation on a GPU”. In: *Journal of Graphics Tools* 16.4 (2012), pp. 218–232 (cit. on p. 13).
- [145] *Thrust: The C++ Parallel Algorithms Library*. <https://nvidia.github.io/thrust>. 2023 (cit. on p. 13).
- [146] J. Nocedal and S. J. Wright. *Numerical Optimization*. New York, New York, US: Springer, 2006 (cit. on pp. 13, 14).
- [147] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004 (cit. on p. 13).
- [148] I. S. Duff, A. M. Erisman, and J. K. Reid. *Direct methods for sparse matrices*. Oxford University Press, 2017 (cit. on p. 13).
- [149] T. A. Davis. *Direct methods for sparse linear systems*. SIAM, 2006 (cit. on p. 13).
- [150] M. R. Hestenes and E. Stiefel. “Methods of conjugate gradients for solving linear systems”. In: *Journal of Research of the National Bureau of Standards* 49.6 (1952), pp. 409–436 (cit. on p. 13).
- [151] R. S. Dembo and T. Steihaug. “Truncated-Newton algorithms for large-scale unconstrained optimization”. In: *Mathematical Programming* 26.2 (1983), pp. 190–212 (cit. on p. 14).
- [152] S. C. Eisentat and H. F. Walker. “Choosing the forcing terms in an inexact Newton method”. In: *SIAM Journal on Scientific Computing* 17.1 (1996), pp. 16–32 (cit. on p. 14).
- [153] T. Bui-Thanh et al. “Extreme-scale UQ for Bayesian inverse problems governed by PDEs”. In: *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*. IEEE. 2012, pp. 1–11 (cit. on p. 15).
- [154] T. Bui-Thanh et al. “A computational framework for infinite-dimensional Bayesian inverse problems Part I: The linearized case, with application to global seismic inversion”. In: *SIAM Journal on Scientific Computing* 35.6 (2013), A2494–A2523 (cit. on p. 15).
- [155] A. Alexanderian et al. “A fast and scalable method for A-optimal design of experiments for infinite-dimensional Bayesian nonlinear inverse problems”. In: *SIAM Journal on Scientific Computing* 38.1 (2016), A243–A272 (cit. on p. 15).
- [156] S. S. Adavani and G. Biros. “Multigrid algorithms for inverse problems with linear parabolic PDE constraints”. In: *SIAM Journal on Scientific Computing* 31.1 (2008), pp. 369–397 (cit. on p. 15).
- [157] G. Biros and G. Doğan. “A multilevel algorithm for inverse problems with elliptic PDE constraints”. In: *Inverse Problems* 24.1–18 (2008) (cit. on p. 15).
- [158] L. Giraud, D. Ruiz, and A. Touhami. “A comparative study of iterative solvers exploiting spectral information for SPD systems”. In: *SIAM Journal on Scientific Computing* 27.5 (2006), pp. 1760–1786 (cit. on p. 15).
- [159] B. Kaltenbacher. “V-cycle convergence of some multigrid methods for ill-posed problems”. In: *Mathematics of Computation* 72.244 (2003), pp. 1711–1730 (cit. on p. 15).
- [160] B. Kaltenbacher. “On the regularizing properties of a full multigrid method for ill-posed problems”. In: *Inverse Problems* 17.4 (2001), pp. 767–788 (cit. on p. 15).
- [161] J. T. King. “On the construction of preconditioners by subspace decomposition”. In: *Journal of Computational and Applied Mathematics* 29 (1990), pp. 195–205 (cit. on p. 15).
- [162] W. Griggs, V. E. Henson, and S. F. McCormick. *A multigrid tutorial*. SIAM, 2000 (cit. on p. 16).

- [163] O. Axelsson and P. S. Vassilevski. “A black box generalized conjugate gradient solver with inner iterations and variable step preconditioning”. In: *SIAM Journal on Matrix Analysis and its Applications* 12.4 (1991), pp. 625–644 (cit. on p. 16).
- [164] Y. Notay. “Flexible conjugate gradients”. In: *SIAM Journal on Scientific Computing* 22.4 (2000), pp. 1444–1460 (cit. on p. 16).
- [165] M. Gutknecht and S. Röllin. “The Chebyshev iteration revisited”. In: *Parallel Computing* 28.2 (2002), pp. 263–283 (cit. on p. 16).
- [166] G. H. Golub and R. S. Varga. “Chebyshev semi-iterative methods, successive overrelaxation iterative methods, and second order Richardson iterative methods”. In: *Numerische Mathematik* 3.1 (1961), pp. 147–156 (cit. on p. 16).
- [167] C. R. Vogel. *Computational methods for inverse problems*. Philadelphia, Pennsylvania, US: SIAM, 2002 (cit. on p. 17).
- [168] E. Haber and J. Modersitzki. “A multilevel method for image registration”. In: *SIAM Journal on Scientific Computing* 27.5 (2006), pp. 1594–1607 (cit. on p. 17).
- [169] E. Haber, U. M. Ascher, and D. Oldenburg. “On optimization techniques for solving nonlinear inverse problems”. In: *Inverse Problems* 16 (2000), pp. 1263–1280 (cit. on p. 17).
- [170] J. Wang and M. Zhang. “Deep Learning for Regularization Prediction in Diffeomorphic Image Registration”. In: *Journal of Machine Learning for Biomedical Imaging* 17 (2021), pp. 1–20 (cit. on p. 18).
- [171] E. Al Safadi and X. Song. “Learning-based image registration with meta-regularization”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 10928–10937 (cit. on p. 18).
- [172] G. E. Christensen et al. “Introduction to the non-rigid image registration evaluation project”. In: *Proc Biomedical Image Registration*. Vol. LNCS 4057. 2006, pp. 128–135 (cit. on p. 18).
- [173] A. Gholami, A. Mang, and G. Biros. “An inverse problem formulation for parameter estimation of a reaction-diffusion model of low grade gliomas”. In: *Journal of Mathematical Biology* 72.1 (2016), pp. 409–433 (cit. on p. 24).
- [174] K. Scheufele et al. “Image-driven biophysical tumor growth model calibration”. In: *SIAM Journal on Scientific Computing* 42.3 (2020), B549–B580 (cit. on p. 24).
- [175] K. Scheufele et al. “Coupling brain-tumor biophysical models and diffeomorphic image registration”. In: *Computer Methods in Applied Mechanics and Engineering* 347 (2019), pp. 533–567 (cit. on p. 24).
- [176] A. Mang et al. “SIBIA-GIS: Scalable biophysics-based image analysis for glioma segmentation”. In: *Proc BraTS 2017 Workshop (MICCAI)*. 2017, pp. 197–204 (cit. on p. 24).
- [177] A. Gooya et al. “GLISTR: Glioma image segmentation and registration”. In: *Medical Imaging, IEEE Transactions on* 31.10 (2013), pp. 1941–1954 (cit. on p. 24).
- [178] A. Mang et al. “Integrated biophysical modeling and image analysis: Application to neuro-oncology”. In: *Annual Review of Biomedical Engineering* 22 (2020), pp. 309–341 (cit. on p. 24).
- [179] C. Hoguea, C. Davatzikos, and G. Biros. “Brain-tumor interaction biophysical models for medical image registration”. In: *SIAM Journal on Imaging Sciences* 30.6 (2008), pp. 3050–3072 (cit. on p. 24).
- [180] E. I. Zacharaki et al. “A comparative study of biomechanical simulators in deformable registration of brain tumor images”. In: *Biomedical Engineering, IEEE Transactions on* 55.3 (2008), pp. 1233–1236 (cit. on p. 24).
- [181] E. I. Zacharaki et al. “Non-diffeomorphic registration of brain tumor images by simulating tissue loss and tumor growth”. In: *NeuroImage* 46.3 (2009), pp. 762–774 (cit. on p. 24).
- [182] C. Hoguea, C. Davatzikos, and G. Biros. “An image-driven parameter estimation problem for a reaction-diffusion glioma growth model with mass effects”. In: *Journal of Mathematical Biology* 56.6 (2008), pp. 793–825 (cit. on p. 24).

- [183] H.-W. Hsieh and N. Charon. “Weight metamorphosis of varifolds and the LDDMM–Fisher–Rao metric”. In: *Calculus of Variations and Partial Differential Equations* 61.5 (2022), p. 165 (cit. on p. 24).
- [184] X. Li et al. “Registration of images with varying topology using embedded maps”. In: *Medical Imaging, IEEE Transactions on* 31.3 (2012), pp. 749–765 (cit. on p. 24).
- [185] A. François et al. “Weighted Metamorphosis for registration of images with different topologies”. In: *International Workshop on Biomedical Image Registration*. Springer. 2022, pp. 8–17 (cit. on p. 24).
- [186] P.-L. Antonsanti et al. “Partial matching in the space of varifolds”. In: *International Conference on Information Processing in Medical Imaging*. Springer. 2021, pp. 123–135 (cit. on p. 24).
- [187] Y. Sukurdeep, M. Bauer, and N. Charon. “A new variational model for shape graph registration with partial matching constraints”. In: *SIAM Journal on Imaging Sciences* 15.1 (2022), pp. 261–292 (cit. on p. 24).
- [188] S. Angenent, S. Haker, and A. Tannenbaum. “Minimizing flows for the Monge–Kantorovich problem”. In: *SIAM Journal on Mathematical Analysis* 35.1 (2003), pp. 61–97 (cit. on p. 26).
- [189] Y. Chen et al. “An efficient algorithm for matrix-valued and vector-valued optimal mass transport”. In: *Journal of Scientific Computing* 77 (2018), pp. 79–100 (cit. on p. 26).

DEPARTMENT OF MATHEMATICS, UNIVERSITY OF HOUSTON, 3551 CULLEN BLVD, HOUSTON, TX 77204-3008
 Email address: andreas@math.uh.edu