Diffeomorphic Iterative Centroid Methods for Template Estimation on Large Datasets

Claire Cury, Joan Alexis Glaunès, Olivier Colliot

Diffeomorphic Iterative Centroid Methods for Template Estimation on Large Datasets

Claire Cury[1234567], Joan A. Glaunès[6], and Olivier Colliot[12345]

[1] Université Pierre et Marie Curie-Paris 6, Centre de Recherche de l'Institut du Cerveau et de la Moëlle épinière, Paris, France
[2] CNRS, UMR 7225, Paris, France
[3] Inserm, UMR-S975, Paris, France
[4] ICM, Institut du Cerveau et de la Moëlle épinière, 47 bd de l'hôpital, Paris, France
[5] Aramis project-team, Inria Paris-Rocquencourt, Paris, France
[6] MAP5, Université Paris Descartes, Sorbonne Paris Cité, France
[7] Corresponding author: claire.cury.pro@gmail.com

Abstract. A common approach for analysis of anatomical variability relies on the estimation of a template representative of the population. The Large Deformation Diffeomorphic Metric Mapping is an attractive framework for that purpose. However, template estimation using LDDMM is computationally expensive, which is a limitation for the study of large datasets. This paper presents an iterative method which quickly provides a centroid of the population in the shape space. This centroid can be used as a rough template estimate or as initialization of a template estimation method. The approach is evaluated on datasets of real and synthetic hippocampi segmented from brain MRI. The results show that the centroid is correctly centered within the population and is stable for different orderings of subjects. When used as an initialization, the approach allows to substantially reduce the computation time of template estimation.

1 Introduction

Large imaging datasets are being increasingly used in neuroscience, thanks to the wider availability of neuroimaging facilities, the development of computing infrastructures and the emergence of large-scale multi-center studies. Such large-scale datasets offer increased statistical power which is crucial for addressing questions such as the relationship between anatomy and genetics or the discovery of new biomarkers using machine learning techniques for instance.

Computational anatomy aims at developing tools for the quantitative analysis of variability of anatomical structures, its variation in healthy and pathological cases and relations between functions and structures [1]. A common approach in computational anatomy is template-based analysis, where the idea is to compare anatomical objects by analyzing their variations relatively to a common template. These variations are analyzed using the ambient space deformations that match each individual structure to the template.

A common requirement is that transformations must be diffeomorphic in order to preserve the topology and to consistently transform coordinates. The Large Deformation Diffeomorphic Metric Mapping (LDDMM) framework [2][3] has been widely used for the study of the geometric variation of human anatomy, of intra-population variability and inter-population differences. It focuses the study on the spatial transformations which can match subject's anatomies one to another, or one to a template structure which needs to be estimated. These transformations not only provide a diffeomorphic correspondence between shapes, but also define a metric distance in shape space.

Several methods have been proposed to estimate templates in the LDDMM framework [4–8]. Vaillant et al. proposed in 2004 [4] a method based on geodesic shooting which iteratively updates a shape by shooting towards the mean of directions of deformations from this shape to all shapes of the population. The method proposed by Glaunès and Joshi [5] starts from the whole population and estimates a template by co-registering all subjects. The method uses a backward scheme: deformations are defined from the subjects to the template. The method optimizes at the same time the deformations between subjects and the template, and the template itself. The template is composed, in the space of currents (more details on section 2.2), by all surfaces of the population. A different approach was proposed by Durrleman et al. [6, 7]. The method initializes the template with a standard shape, in practice it is often an ellipsoid. The method uses a forward scheme: deformations are defined from the template to the subjects. Again, it optimizes at the same time the deformations and the template. The template is composed by one surface which presents the same configuration as the initial ellipsoid. The method presented by Ma et al. [8] introduces an hyper template which is an extra fixed shape (which can be a subject of the population). The method aims at optimizing at the same time deformations from the hyper template to the template and deformations from the template to subjects of the population. The template is optimized via the deformation of the hyper template, not directly.

A common point of all these methods is that they need a surface matching algorithm, which is very expensive in terms of computation time in the LDDMM framework. When no specific optimization is used, computing only one matching between two surfaces, each composed of 3000 vertices, takes approximately 30 to 40 minutes. Then, computing a template composed of one hundred such surfaces until convergence can take a few days or some weeks. This is a limitation for the study of large databases. Different strategies can be used to reduce computation time. GPU implementation can substantially speed up the computation of convolutions that are heavily used in LDDMM deformations. Matching pursuit on current can also be used to reduce the computation time [9]. Sparse representations of deformations allow to reduce the number of optimized parameters of the deformations [7].

Here, we propose a new approach to reduce the computation time called diffeomorphic iterative centroid using currents. The method provides in N-1 steps (with N the number of shapes of the population) a centroid already correctly

centered within the population of shapes. It increases the convergence speed of the template estimation by providing an initialization that is closer to the target.

Our method has some close connections with more general iterative methods to compute means on Riemannian manifolds. For example Arnaudon et al. [10] defined a stochastic iterative method which converges to the Fréchet mean of the set of points. Ando, Li and Mathias [11] gave a recursive definition of the mean of positive definite matrices which verifies important properties of geometric means. However these methods require a large number of iterations (much larger than the number of points of the dataset), while in our case, due to the high computational cost of matchings, we aim at limiting as much as possible the number of iterations.

The paper is organized as follows. First, we present the mathematical framework of LDDMM and currents (Section 2). Section 3.1 then introduces the template estimation and the iterative centroid method. In Section 4.1, we evaluate the approach on datasets of real and synthetic hippocampi extracted from brain magnetic resonance images (MRI).

2 Notation and mathematical setup

For the Diffeomorphic Iterative Centroid method, we use the LDDMM framework (2.1) to quantify the difference between shapes. To model surfaces of the population, we use the framework of currents (2.2) which does not assume point-to-point correspondences.

2.1 Large Diffeomorphic Deformations.

The Large Diffeomorphic Deformation Metric Mapping framework allows quantifying the difference between shapes and provides a shape space representation: shapes of the population are seen as points in an infinite dimensional smooth manifold, providing a continuum between shapes of the population. In this framework a diffeomorphism deforms the whole space, not only a shape.

Diffeomorphisms as flows of vector fields. In the LDDMM framework, deformation maps $\varphi : \mathbb{R}^3 \to \mathbb{R}^3$ are generated by integration of time-dependent vector fields $v(x,.)$ in an Hilbert space V, with $x \in \mathbb{R}^3$ and $t \in [0,1]$. If $v(x,t)$ is regular enough, i.e. if we consider the vector fields $(v(\cdot,t))_{t\in[0,1]}$ in $L^2([0,1],V)$, where V is a Reproducing Kernel Hilbert Space (R.K.H.S.) embedded in the space of $C^1(\mathbb{R}^3,\mathbb{R}^3)$ vector fields vanishing at infinity, then the transport equation:

$$\begin{cases} \frac{d\phi_v}{dt}(x,t) = v(\phi_v(x,t),t) & \forall t \in [0,1] \\ \phi_v(x,0) = x & \forall x \in \mathbb{R}^3 \end{cases} \tag{1}$$

has a unique solution, and one sets $\varphi_v = \phi_v(\cdot,1)$ the diffeomorphism induced by $v(x,t)$. The induced set of diffeomorphisms $\mathcal{A}_V$ is a subgroup of the group

of C^1 diffeomorphisms. To enforce velocity fields to stay in this space, one must control the energy

$$E(v) := \int_0^1 \|v(\cdot,t)\|_V^2 \ dt. \tag{2}$$

Metric structure on the diffeomorphisms group The induced subgroup of diffeomorphisms $\mathcal{A}_V$ is equipped with a right-invariant metric defined by the rules: $\forall \varphi, \psi \in \mathcal{A}_V$,

$$\begin{cases} D(\varphi,\psi) = D(Id, \varphi^{-1} \circ \psi) \\ D(Id,\varphi) = \inf \left\{ \int_0^1 \|v(\cdot,t)\|_V \ dt \ ; \ v \in L^2([0,1],V), \varphi_v = \varphi \right\} \end{cases} \tag{3}$$

$D(\varphi,\psi)$ represents the shortest length of paths connecting φ to ψ in the diffeomorphisms group. Moreover, as in the classical Riemannian theory, minimizing the length of paths is equivalent to minimizing their energy, and one has also:

$$D(Id,\varphi)) = \inf \left\{ E(v) \ ; \ v \in L^2([0,1],V), \varphi_v = \varphi \right\} \tag{4}$$

Discrete matching functionals. Considering two surfaces S and T, the optimal matching between them is defined in an ideal setting, as the map φ_v minimizing $E(v)$ under the constraint $\varphi_v(S) = T$. In practice such an exact matching is often not feasible and one writes inexact unconstrained matching functionals which minimize both $E(v)$ and a matching criterion which evaluates the spatial proximity between $\varphi_v(S)$ and T, as we will see in the next section.

In a discrete setting, when the matching criterion depends only on φ_v via the images $\varphi_v(x_i)$ of a finite number of points x_i (such as the vertices of the mesh S) one can show that the vector fields $v(x,t)$ which induce the optimal deformation map can be written via a convolution formula over the surface involving the reproducing kernel K_V of the R.K.H.S. V. This is due to the reproducing property of V; indeed V is the closed span of vectors fields of the form $K_V(x,.)\alpha$, and therefore $v(x,t)$ writes

$$v(x,t) = \sum_{i=1}^{n} K_V(x, x_i(t))\alpha_i(t), \tag{5}$$

where $x_i(t) = \phi_v(x_i,t)$ are the trajectories of points x_i, and $\alpha_i(t) \in \mathbb{R}^3$ are time-dependent vectors called momentum vectors, which parametrize completely the deformation. Trajectories $x_i(t)$ depend only on these vectors as solutions of the following system of ordinary differential equations:

$$\frac{dx_j(t)}{dt} = \sum_{i=1}^{n} K_V(x_j(t), x_i(t))\alpha_i(t), \tag{6}$$

for $1 \leq j \leq n$. This is obtained by plugging formula 5 for the optimal velocity fields into the flow equation 1 taken at $x = x_j$. Moreover, the energy $E(v)$ takes

an explicit form as expressed in terms of trajectories and momentum vectors:

$$E(v) = \int_0^1 \sum_{i,j=1}^{n} \alpha_i(t)^T K_V(x_i(t), x_j(t))\alpha_j(t)\, dt. \tag{7}$$

These equations reformulate the problems in a finite dimensional Riemannian setting. Indeed $E(v)$ appears as the energy of the path $t \mapsto (x_i(t))_{1\leq i\leq n}$ in the space of landmarks $\mathcal{L}^n = \{\mathbf{x} = (x_i)_{1\leq i\leq n}, x_i \neq x_j \quad \forall i,j\}$ equipped with local metric $g(\mathbf{x}) = K(\mathbf{x})^{-1}$, where $K_V(\mathbf{x})$ is the $3n \times 3n$ matrix with block entries $K_V(x_i, x_j)$, $1 \leq i, j \leq n$.

Geodesic equations and local encoding. As introduced previously, the minimization of the energy $E(v)$ in matching problems can be interpreted as the estimation of a length-minimizing path in the group of diffeomorphisms $\mathcal{A}_V$, and also additionally as a length-minimizing path in the space of landmarks when considering discrete problems. Such length-minimizing paths obey some geodesic equations [4] (distances are define as in 3), which write as follows in the case of landmarks (using matrix notations):

$$\begin{cases} \frac{d\mathbf{x}(t)}{dt} = K_V(\mathbf{x}(t))\boldsymbol{\alpha}(t) \\ \frac{d\boldsymbol{\alpha}(t)}{dt} = -\frac{1}{2}\nabla_{\mathbf{x}(t)}\left[\boldsymbol{\alpha}(t)^T K_V(\mathbf{x}(t))\boldsymbol{\alpha}(t)\right], \end{cases} \tag{8}$$

Note that the first equation is nothing more than equation 6 which allows to compute trajectories $x_i(t)$ from any time-dependant momentum vectors $\alpha_i(t)$, while the second equation gives the evolution of the momentum vectors themselves. This new set of ODEs can be solved from any initial conditions $x_i(0), \alpha_i(0)$, which means that the initial momentum $\alpha_i(0)$ fully determine the subsequent time evolution of the system (since the $x_i(0)$ are fixed points). As a consequence, these initial momentum vectors encode all information of the optimal diffeomorphism. This is a very important point for applications, specifically for group studies, since it allows to analyse the set of deformation maps from a given template to the observed shapes by performing statistics on the initial momentum vectors located on the template shape. We also can use geodesic shooting from initial conditions $x_i(0)\alpha_i(0)$ in order to generate any arbitrary deformation of a shape in the shape space. We will use this tool for the construction of our synthetic dataset Data1 (see section 4.1).

2.2 Currents

The idea of the mathematical object named "currents" is related to the theory of distributions as presented by Schwartz in 1952 [12], in which distributions are characterized by their action on any smooth functions with compact support. In 1955, De Rham [13] generalized distributions to differential forms to represent submanifolds, and called this representation currents. This mathematical object serves to model geometrical objects using a non parametric representation.

The use of currents in computational anatomy was introduced by J. Glaunés and M. Vaillant in 2005 [14][15] and subsequently developed by Durrleman ([16]) in order to provide a dissimilarity measure between meshes which does not assume point-to-point correspondence between anatomical structures. The approach proposed by Vaillant and Glaunès is to represent meshes as objects in a linear space and supply it with a computable norm. Using currents to represent surfaces has some benefits. First it avoids the point correspondence issue: one does not need to define pairs of corresponding points between two surfaces to evaluate their spatial proximity. Moreover, metrics on currents are robust to different samplings and topologies and take into account not only the global shapes but also their local orientations. Another important benefit is that the space of currents is a vector space, which allows to consider linear combinations such as means of shapes in the space of currents. This property will be used in the centroid and template methods that we introduce in the following.

We limit the framework to surfaces embedded in $\mathbb{R}^3$. Let S be an oriented compact surface, possibly with boundary. Any smooth and compactly supported differential 2-form ω of $\mathbb{R}^3$ - i.e. a mapping $x \mapsto \omega(x)$ such that for any $x \in \mathbb{R}^3$, $\omega(x)$ is a 2-form, an alternated bilinear mapping from $\mathbb{R}^3 \times \mathbb{R}^3$ to $\mathbb{R}$ - can be integrated over S

$$\int_S \omega = \int_S \omega(x)(u_1(x), u_2(x))d\sigma(x). \tag{9}$$

where $(u_1(x), u_2(x))$ is an orthonormal basis of the tangent plane at point x, and $d\sigma$ the Lebesgue measure on the surface S. Hence one can define a linear form $[S]$ over the space of 2-forms via the rule $[S](\omega) := \int_S \omega$. If one defines a Hilbert metric on the space of 2-forms such that the corresponding space is continuously embedded in the space of continuous bounded 2-forms, this mapping will be continuous [14], which will make $[S]$ an element of the space of 2-currents, the dual space to the space of 2-forms.

Note that since we are working with 2-forms on $\mathbb{R}^3$, we can use a vectorial representation via the cross product: for every 2-form ω and $x \in \mathbb{R}^3$ there exists a vector $\overline{\omega}(x) \in \mathbb{R}^3$ such that for every $\alpha, \beta \in \mathbb{R}^3$,

$$\omega(x)(\alpha, \beta) = \langle \overline{\omega}(x)\ ,\ \alpha \times \beta \rangle = det(\alpha, \beta, \overline{\omega}(x)), \tag{10}$$

Therefore we can work with vector fields $\overline{\omega}$ instead of 2-forms ω. In the following, with a slight abuse of notation, we will use $\omega(x)$ to represent both the bilinear alternated form and its vectorial representative. Hence the current of a surface S can be re-written from equation 9 as follows:

$$[S](\omega) = \int_S \langle \omega(x)\ ,\ n(x) \rangle\ d\sigma(x) \tag{11}$$

with $n(x)$ the unit normal vector to the surface: $n(x) := u_1(x) \times u_2(x)$.

We define a Hilbert metric $\langle \cdot\ ,\ \cdot \rangle_W$ on the space of vector fields of $\mathbb{R}^3$, and require the space W to be continuously embedded in $C_0^1(\mathbb{R}^3, \mathbb{R}^3)$. The space of

currents we consider is the space of continuous linear forms on W, i.e. the dual space W^*, and the required embedding property ensures that for a large class of oriented surfaces S in $\mathbb{R}^3$, comprising smooth surfaces and also triangulated meshes, the associated linear mapping $[S]$ is indeed a current, i.e. it belongs to W^*.

The central object from the computational point of view is the reproducing kernel of space W, which we introduce here. For any point $x \in \mathbb{R}^3$ and vector $\alpha \in \mathbb{R}^3$ one can consider the Dirac functional $\delta_x^\alpha : \omega \mapsto \langle \omega(x)\ ,\ \alpha \rangle$ which is an element of W^*. The Riesz representation theorem then states that there exists a unique $u \in W$ such that for all $\omega \in W$, $\langle u\ ,\ \omega \rangle_W = \delta_x^\alpha(\omega) = \langle \omega(x)\ ,\ \alpha \rangle$. u is thus a vector field which depends on x and linearly on α, and we write it $u = K_W(\cdot, x)\alpha$. Thus we have the rule

$$\langle K_W(\cdot, x)\alpha, \omega \rangle_W = \langle \omega(x)\ ,\ \alpha \rangle\,. \tag{12}$$

Moreover, applying this formula to $\omega = K_W(\cdot, y)\beta$ for any other point $y \in \mathbb{R}^3$ and vector $\beta \in \mathbb{R}^3$, we get

$$\begin{aligned} \langle K_W(\cdot, x)\alpha, K_W(\cdot, y)\beta \rangle_W &= \langle K_W(x, y)\beta\ ,\ \alpha \rangle \\ &= \alpha^T K_W(x, y)\beta = \left\langle \delta_x^\alpha\ ,\ \delta_y^\beta \right\rangle_{W^*}. \end{aligned} \tag{13}$$

$K_W(x, y)$ is a 3×3 matrix, and the mapping $K_W : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}^{3\times 3}$ is called the reproducing kernel of the space W. Now, note that we can rewrite equation 11 as

$$[S](\omega) = \int_S \delta_x^{n(x)}(\omega)\ d\sigma(x) \tag{14}$$

Thus using equation 13, one can prove that for two surfaces S and T,

$$\langle [S]\ ,\ [T] \rangle_{W^*}^2 = \int_S \int_T \langle n_S(x)\ ,\ K_W(x, y)n_T(y) \rangle\ ds(x)ds(y) \tag{15}$$

This formula defines the metric we use for evaluating spatial proximity between shapes. It is clear that the type of kernel one uses fully determines the metric and therefore will have a direct impact on the behaviour of the algorithms. We use scalar invariant kernels of the form $K_W(x, y) = h(\|x - y\|^2/\sigma_W^2)I_3$, where h is a real function such as $h(r) = e^{-r}$ (gaussian kernel) or $h(r) = 1/(1 + r)$ (Cauchy kernel), and σ_W a scale factor. In practice this scale parameter has a strong influence on the results; we will go back to this point later.

We can now define the optimal match between two currents $[S]$ and $[T]$, which is the diffeomorphism minimizing the functional

$$J_{S,T}(v) = \gamma E(v) + \|[\varphi_v(S)] - [T]\|_{W^*}^2 \tag{16}$$

This functional is non convex and in practice we use a gradient descent algorithm to perform the optimization, which cannot guarantee to reach a global minimum. We observed empirically that local minima can be avoided by using a multi-scale approach in which several optimization steps are performed with decreasing

values of the width σ_W of the kernel K_W (each step provides an initial guess for the next one).

In practice, surfaces are given as triangulated meshes, which we discretize in the space of currents W^* by combinations of Dirac functionals : $[S] \simeq \sum_{f \in S} \delta_{c_f}^{n_f}$, where the sum is taken over all triangles $f = (f^1, f^2, f^3)$ of the mesh S, and $c_f = \frac{1}{2}(f^1 + f^2 + f^3)$, $n_f = \frac{1}{2}(f^2 - f^1) \times (f^3 - f^1)$ denote respectively the center and normal vector of the triangle. Given a deformation map φ and a triangulated surface S, we also approximate its image $\varphi(S)$ by the triangulated mesh obtained by letting ϕ act only on the vertices of S. This leads us to the following discrete formulation of the matching problem:

$$\begin{aligned} J_{S,T}^d(\alpha) = \gamma \int_0^1 \sum_{i=1}^n \alpha_i(t)^T K_V(x_i(t), x_j(t))\alpha_j(t)\, dt \\ + \sum_{f,f' \in S} n_{\varphi(f)}^T K_W(c_{\varphi(f)}, c_{\varphi(f')}) n_{\varphi(f')} \\ + \sum_{g,g' \in T} n_g K_W(c_g, c_{g'}) n_{g'} - 2 \sum_{f \in S, g \in T} n_{\varphi(f)}^T K_W(c_{\varphi(f)}, c_g) n_g \end{aligned} \quad (17)$$

where φ denotes the diffeomorphism associated to momentum vectors $\alpha_i(t)$ and trajectories $x_i(t)$, $x_i = x_i(0)$ being the vertices of mesh S, and where we have noted for any face f, $\varphi(f) = (\varphi(f^1), \varphi(f^2), \varphi(f^3))$. We note 3 important parameters, γ which controls the regularity of the map, σ_V which controls the scale in the space of deformations and σ_W which controls the scale in the space of currents.

3 A template estimation for large database via LDDMM

3.1 Why building a template?

A central notion in computational anatomy is the generation of registration maps, mapping a large set of anatomical data to a common coordinate system to study intra-population variability and inter-population differences. In this paper, we use the method introduced by Glaunès et al. [5] which estimates a template given a collection of unlabelled points sets or surfaces in the framework of scalar measures and currents. In our case we use the framework of currents. This method is posed as a minimum mean squared error estimation problem and uses the metric on the space of diffeomorphisms. Let S_i be N surfaces in $\mathbb{R}^3$ (i.e. the whole surface population). Let $[S_i]$ be the corresponding current of S_i, or its approximation by a finite sum of vectorial Diracs. The problem is formulated as follows:

$$\left\{\hat{v_i}, \hat{\mathcal{T}}\right\} = \arg\min_{v_i, \mathcal{T}} \sum_{i=1}^N \left\{ \| \mathcal{T} - [\varphi_{v_i}(S_i)] \|_{W^*}^2 + \gamma E(v_i) \right\}, \quad (18)$$

where the minimization is performed over the spaces $L^2([0,1], V)$ for the velocity fields v_i and over the space of currents W^* for $\mathcal{T}$. The method uses an alternated

optimization i.e. surfaces are successively matched to the template, then the template is updated and this sequence is iterated until convergence. One can observe that when φ_i is fixed, the functional is minimized when $\mathcal{T}$ is the average of $[\varphi_i(S_i)]$ in space W^*:

$$\mathcal{T} = \frac{1}{N} \sum_{i=1}^{N} [\varphi_{v_i}(S_i)] \,, \tag{19}$$

which makes the optimization with respect to $\mathcal{T}$ straightforward. This optimal current is not a surface itself; in practice it is constituted by the union of all surfaces $\varphi_{v_i}(S_i)$, and the $\frac{1}{N}$ factor acts as if all normal vectors to these surfaces were weighted by $\frac{1}{N}$. At the end of the optimization process however, all surfaces being co-registered, the $\hat{\varphi}_{v_i}(S_i)$ are close to each other, which makes the optimal template $\hat{\mathcal{T}}$ close to being a true surface.

In practice, we stop the template estimation method after P loops, and with the datasets we use, $P = 7$ seems to be sufficient to obtain an adequate template.

As detailed in section 2, obtaining a template allows to perform statistical analysis of the deformation maps via the initial momentum representation to characterize the population. One can run analysis on momentum vectors such as Principal Component Analysis (PCA), or estimate an approximation of pairwise diffeomorphic distances between subjects using the estimated template (Yang et al [17]) in order to use manifold learning methods like Isomap [18].

In the present case, the optimal template for the population is not a true surface but is defined, in the space of currents, by the mean $\hat{\mathcal{T}} = \frac{1}{N} \sum_{j=1}^{N} \hat{\varphi}_{v_j} [S_j]$. However this makes no difference from the point of view of statistical analysis, because this template can be used in the LDDMM framework exactly as if it was a true surface.

One may speed up the estimation process and avoid local minima issues by defining a good initialization of the optimization process. Standard initialization consists in setting $\mathcal{T} = \frac{1}{N} \sum_{i=1}^{N} [S_i]$, which means that the initial template is defined as the combination of all unregistered shapes in the population. Alternatively, if one is given a good initial guess $\mathcal{T}$, the convergence speed of the method can be improved. This is the primary motivation for the introduction of the iterative centroid method which we present in the next section.

3.2 The Iterative Centroid method

As presented in the introduction, computing a template in the LDDMM framework can be highly time consuming, taking a few days or some weeks for large real-world databases. To increase the speed of the method, one of the key points may be to start with a good initial template, already correctly centred among shapes in the population. Of course the computation time of such an initialization method must be substantially lower than the template estimation itself. The Iterative Centroid method presented here performs such an initialization with $N - 1$ pairwise matchings only.

The LDDMM framework, in an ideal setting (exact matching between shapes), sets the template estimation problem as the computation of a centroid on a Riemannian manifold, which is of finite dimension in the discrete case (we limit our analysis to this finite dimensional setting in what follows). The Fréchet mean is the standard way for defining such a centroid and provides the basic inspiration of all LDDMM template estimation methods. Since our Iterated Centroid method is also inspired by considerations about computation of centroids in Euclidean space and their analogues on Riemannian manifolds, we will briefly discuss these ideas in the following.

Centroid computation on Euclidean and Riemannian spaces. If $x_i, 1 \leq i \leq N$ are points in $\mathbb{R}^d$, then their centroid is defined as

$$b_N = \frac{1}{N} \sum_{i=1}^{N} x_i. \tag{20}$$

It satisfies also the following:

$$b_N = \arg\min_{y \in \mathbb{R}^d} \sum_{1 \leq i \leq N} \|y - x_i\|^2. \tag{21}$$

Now, when considering points x_i living on a Riemannian manifold M (we assume M is path-connected and geodesically complete), the definition of b_N cannot be used because M is not a vector space. However the variational characterization of b_N has an analogue, which leads to the definition of the Fréchet mean, also called 2-mean, which is uniquely defined under some constraints (see [10]) on the relative locations of points x_i in the manifold:

$$b_N = \arg\min_{y \in M} \sum_{1 \leq i \leq N} d_M(y, x_i)^2. \tag{22}$$

Many mathematical studies (as for example Kendall [19], Karcher [20] Le [21], Afsari [22, 23]), have focused on proving the existence and uniqueness of the mean, as well as proposing algorithms to compute it. The more general notion of p-mean of a probability measure μ on a Riemannian manifold M is defined by:

$$b = \arg\min_{x \in M} F_p(x), \quad F_p(x) = \int_M d_M(x, y)^p \mu(dy). \tag{23}$$

Arnaudon et al. [10] published in 2012 for $p \geq 1$ a stochastic algorithm which converges almost surely to the p-mean of the probability measure μ. This algorithm does not require to compute the gradient of the functional F_p to minimize. The authors construct a time inhomogeneous Markov chain by choosing at each step a random point P with distribution μ and moving the current point X to a new position along the geodesic connecting X to P. As it will be obvious in the following, our method shares similarities with this method for the case $p = 2$, in

that it also uses an iterative process which at each step moves the current position to a new position along a geodesic. However our method is not stochastic and does not compute the 2-mean of the points. Moreover, our approach stops after $N-1$ iterations, while on the contrary the stochastic method does not ensure to have considered all subjects of the population after N iterations.

Other definitions of centroids in the Riemannian setting can be proposed. The following ideas are more directly connected to our method. Going back to the Euclidean case, one can observe that b_N satisfies the following iterative relation:

$$\begin{cases} b_1 = x_1 \\ b_{k+1} = \frac{k}{k+1} b_k + \frac{1}{k+1} x_{k+1}, \quad 1 \leq k \leq N-1, \end{cases} \tag{24}$$

which has the side benefit that at each step b_k is the centroid of the $x_i, 1 \leq i \leq k$. This iterative process has an analogue in the Riemannian case, because one can interpret the convex combination $\frac{k}{k+1} b_k + \frac{1}{k+1} x_{k+1}$ as the point located along the geodesic linking b_k to x_{k+1}, at a distance equal to $\frac{1}{k+1}$ of the total length of the geodesic, which we can write $geod(b_k, x_{k+1}, \frac{1}{k+1})$. This leads to the following definition in the Riemannian case:

$$\begin{cases} \tilde{b}_1 = x_1 \\ \tilde{b}_{k+1} = geod(\tilde{b}_k, x_{k+1}, \frac{1}{k+1}), \quad 1 \leq k \leq N-1, \end{cases} \tag{25}$$

Of course this new definition of centroid does not coincide with the Fréchet mean when the metric is not Euclidean, and furthermore it has the drawback to depend on the ordering of points x_i. Moreover one may consider other iterative procedures such as computing midpoints between arbitrary pairs of points x_i, and then midpoints of the midpoints, etc. In other words, all procedures that are based on decomposing the Euclidean equality $b_N = \frac{1}{N} \sum_{i=1}^{N} x_i$ as a sequence of pairwise convex combinations lead to possible alternative definitions of centroid in a Riemannian setting. Based on these remarks, Emery and Mokobodzki [24] proposed to define the centroid not as a unique point but as the set B_N of points $x \in M$ satisfying

$$f(x) \leq \frac{1}{N} \sum_{i=1}^{N} f(x_i), \tag{26}$$

for any convex function f on M (a convex function f on M being defined by the property that its restriction to all geodesics is convex). This set B_N takes into account all centroids obtained by bringing together points x_i by all possible means, i.e. recursively by pairs, or by iteratively adding a new point, as explained above (see Fig.2).

Outline of the method The Iterated Centroid method consists roughly in applying the following procedure: given a collection of N shapes S_i, we successively update the centroid by matching it to the next shape and moving along the geodesic flow. Figure 1 illustrates the general idea. We propose two alternative ways for the update step (algorithms 1 and 2 below).

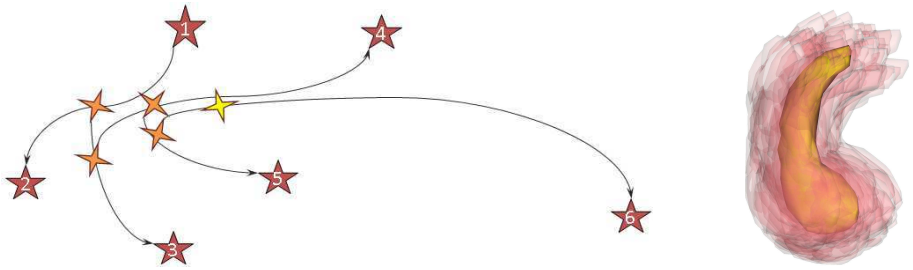

Fig. 1. Illustration of the method. Left image: red stars are subjects of the population, the yellow star is the final Centroid, and orange stars are iterations of the Centroid. Right image: Final Centroid with the hippocampus population from Data1 (red). See section 4.1 for more details about datasets.

Direct Iterative Centroid : IC1 The first version of the method computes a centroid between two objects O_1 and O_2 by transporting a first object O_1 along the geodesic going from this object to O_2. The transport is stopped depending of the weights of the objects. If the weight of O_1 is w_1, and the weight of O_2 is w_2 with $w_1 + w_2 = 1$, we stop the deformation of O_1 at time $t = w_2$. Since the method is iterative, the first two objects are two subjects of the population, for the next step we have as a first object the previous centroid and as a second object a new subject of the population. The algorithm proceeds as presented in the Algorithm 1.

Data: N surfaces S_i
Result: 1 surface B_N representing the centroid of the population
$B_1 = S_1$;
for *i from* 1 *to* $N-1$ **do**
  B_i is matched using the equation (16) to S_{i+1} which results in a deformation map $\phi_{v_i}(x,t)$;
  Set $B_{i+1} = \phi_{v_i}(B_i, \frac{1}{i+1})$ which means we transport B_i along the geodesic and stop at time $t = \frac{1}{i+1}$;
end

Algorithm 1: Iterative Centroid 1 (IC1)

Iterative Centroid with averaging in the space of currents : IC2 Because matchings are inaccurate, the centroid computed with the method presented above accumulates small errors which can have an impact on the final centroid. Furthermore, the centroid computed with algorithm 1 is in fact a deformation of the first shape S_1, which makes the procedure even more dependent on the ordering of subjects than it would be in an ideal exact matching setting. In this second

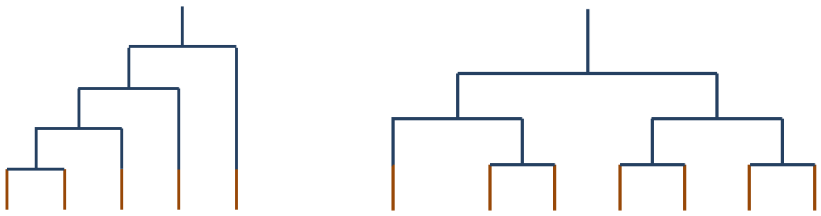

Fig. 2. Diagrams of the iterative processes which lead to the centroid computation. The tops of the diagrams represent the final centroid. The diagram on the left corresponds to the Iterative Centroid algorithms (IC1 and IC2). The diagram on the right corresponds to the pairwise algorithm (PW).

algorithm, we modify the updating step by computing a mean in the space of currents between the deformation of the current centroid and the backward flow of the curent shape being matched. Hence the computed centroid is not a true surface but a current, i.e. combination of surfaces, as in the template estimation method. The weights chosen in the averaging reflects the relative importance of the new shape, so that at the end of the procedure, all shapes forming the centroid have equal weight $\frac{1}{N}$. The algorithm proceeds as presented in Algorithm 2.

Data: N surfaces S_i
Result: 1 current $\mathcal{B}_N$ representing the centroid of the population
$\mathcal{B}_1 = [S_1]$;
for *i from* 1 *to* $N-1$ **do**
  $\mathcal{B}_i$ is matched using the equation (16) to S_{i+1} which results in a deformation map $\phi_{v_i}(x,t)$;
  Set $\mathcal{B}_{i+1} = \frac{i}{i+1} * \phi_{v_i}(\mathcal{B}_i, \frac{1}{i+1}) + \frac{1}{i+1}[\phi_{u_i}(S_{i+1}, \frac{i}{i+1})]$ which means we transport $\mathcal{B}_i$ along the geodesic and stop at time $t = \frac{1}{i+1}$;
  where $u_i(x,t) = -v_i(x, 1-t)$, i.e. ϕ_{u_i} is the reverse flow map.
end

Algorithm 2: Iterative Centroid 2 (IC2)

Note that we have used the notation $\phi_{v_i}(\mathcal{B}_i, \frac{1}{i+1})$ to denote the transport (push-forward) of the current $\mathcal{B}_i$ by the diffeomorphism. Here $\mathcal{B}_i$ is a linear combination of currents associated to surfaces, and the transported current is the linear combination (keeping the weights unchanged) of the currents associated to the transported surfaces.

An alternative method : Pairwise Centroid (PW) Another possibility is to group objects by pairs, compute centroids (middle points) for each pair, and then re-

cursively apply the same procedure to the set of centroids, until having only one centroid (see Fig. 2). This pairwise method also depends on the ordering of subjects, and also provides a centroid which satisfies the definition of Emery and Mokobodzki (disregarding the inaccuracy of matchings).
When the population is composed of more than 3 subjects, we split the population in two parts and recursively apply the same splitting until having two or three objects in each group. We then apply algorithm 1 to obtain the corresponding centroid before going back up along the dyadic tree, and keeping attention to the weight of each object. This recursive algorithm is described in algorithm 3.

Data: N surfaces S_i
Result: 1 surface B representing the centroid of the population
if $N \geq 2$ **then**
  B_{left} = Pairwise Centroid $(S_1, ..., S_{[N/2]})$;
  B_{right} = Pairwise Centroid $(S_{[N/2]+1}, ..., S_N)$;
  B_{left} is matched to B_{right} which results in a deformation map $\phi_v(x,t)$;
  Set $B = \phi_v(B_{left}, \frac{[N/2]+1}{N})$ which means we transport B_{left} along the geodesic and stop at time $t = \frac{[N/2]+1}{N}$;
end
else
  $B = S_1$
end

Algorithm 3: Pairwise Centroid

3.3 Implementation

The methods presented just before need some parameters. Indeed, in each algorithm we have to compute the matching from one surface to another. For each matching we minimize the corresponding functional (see equation 17 at the end of section 2.2) which estimates the news momentum vectors α, which then are used to update the positions of points x_i of the surface. A gradient descent with adaptive step size is used for the minimization of the functional J. Evaluation of the functional and its gradient require numerical integrations of high-dimensional ordinary differential equations, which is done using Euler trapezoidal rule.
The main parameters for computing J are *maxiter* which is the maximum number of iterations for the adaptive step size gradient descent algorithm, γ for the regularity of the matching, and σ_W and σ_V the sizes of the kernels which control the metric of the spaces W and V.

We selected parameters in order to have relatively good matchings in a short time. We chose γ close enough to zero to enforce the matching to bring the first

object to the second one. Nevertheless, we must be prudent: choosing a γ too small could be hazardous because the regularity of the deformation could not be preserved. For each pairwise matching, we use the multi-scale approach described in section 2.2 page 5, performing four consecutive optimization processes with decreasing values by a constant factor of the σ_W parameter which is the size of the R.K.H.S. W, to increase the precision of the matching. At the beginning, we fix this σ_W parameter with a sufficient large value in order the capture the possible important variations or differences between shapes. This is for this reason that for the two first minimizations of the functional, we use a small *maxiter* parameter. For the results presented after, we used very small values for the parameter $maxiter = [50, 50, 100, 300]$, to increase the velocity of the method. Results can be less accurate than in our previous study [25] which used different values for *maxiter*: $[40, 40, 100, 1000]$, which take twice as much time to compute. For the kernel size σ_V of the deformation space, we fix this parameter at the beginning and have to adapt it to the size of the data.

The first method starts from N surfaces, and gives a centroid composed by only one surface, which is a deformation of the surface used at the initialization step. An example is shown in Fig. 3. This method is rather fast, because at each step we have to match only one mesh composed by n_1 vertices to another, where n_1 is the number of vertices of the first mesh of the iterative procedure.

The second method starts from N surfaces and gives a centroid composed of deformations of all surfaces of the population. At each step it forms a combination in the space of currents between the current centroid and a backward flow of the new surface being matched. In practice this implies that the centroid grows in complexity; at step i its number of vertices is $\sum_{j=1}^{i} j * n_j$. Hence this algorithm is slower than the first one, but the mesh structure of the final centroid does not depend on the mesh of only one subject of the population, and the combination compensates the bias introduced by the inaccuracy of matchings.

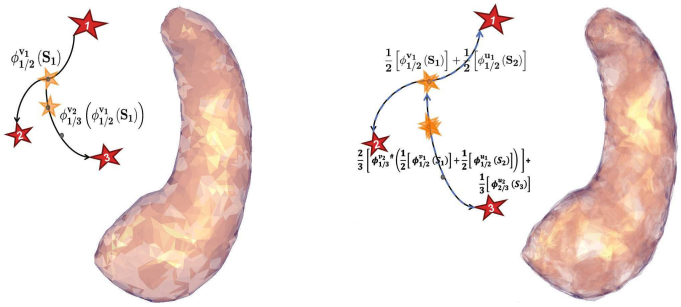

Fig. 3. On the left, an Iterative Centroid of the dataset Data2 (see section 4.1 for more details about datasets) computed using the IC1 algorithm, and on the right the IC2 algorithm.

The results of the Iterative Centroid algorithms depend on the ordering of subjects. We will study this dependence in the experimental part, and also study the effect of stopping the I.C. before it completes all iterations.

4 Experiments and Results

4.1 Data

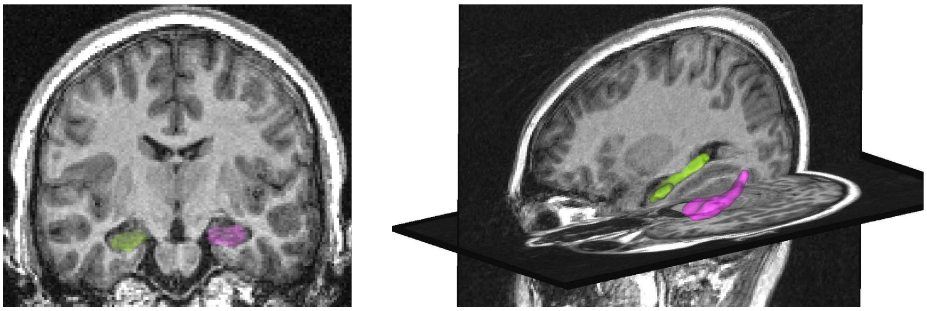

Fig. 4. Left panel: coronal view of the MRI with the meshes of hippocampi segmented by the SACHA software [26], the right hippocampus is in green and the left one in pink. Right panel: 3D view of the hippocampi.

To evaluate our approach, we used data from 95 young (14-16 years old) subjects from the European database IMAGEN. The anatomical structure that we considered was the hippocampus, which is a small bilateral structure of the temporal lobe of the brain involved in memory processes. The hippocampus is one of the first structures to be damaged in Alzheimer's disease; it is also implicated in temporal lobe epilepsy, and is altered in stress and depression. 95 left hippocampi were segmented from T1-weighted Magnetic Resonance Images (MRI) of this database (see Fig. 4) with the software SACHA [26], before computing meshes from the binary masks using BrainVISA software [8].

We denote as RealData the dataset composed of all 95 hippocampi meshes. We rigidly aligned all hippocampi to one subject of the population. For this rigid registration, we used a similarity term based on measures (as in [27]) rather than currents.

We also built two synthetic populations of hippocampi meshes, denoted as Data1 and Data2. Data1 is composed of a large number of subjects, in order to test our algorithms on a large dataset. In order to study separately the effect of the population size, meshes of this population are simple. Data2 is a synthetic population close to the real one, with the difference that all subjects

[8] http://www.brainvisa.info

have the same mesh structure. This allows to test our algorithms in a population with a single mesh structure, thus disregarding the effects of different mesh structures. These two datasets are defined as follows (examples of subjects from these datasets are shown on Fig. 5):

- Data1. We chose one subject S_0 that we decimated (down to 135 vertices) and deformed using geodesic shooting in 500 random directions with a sufficiently large kernel and a reasonable momentum vector norm in order to preserve the overall hippocampal shape, resulting in 500 deformed objects. Each deformed object was then further transformed by a translation and a rotation of small magnitude. This resulted in the 500 different shapes of Data1. All shapes in Data1 have the same mesh structure. Data1 thus provides a large dataset with simple meshes and mainly global deformations.
- Data2. We chose the same initial subject S_0 that we decimated to 1001 vertices. We matched this mesh to each subject of the dataset RealData ($n = 95$), using diffeomorphic deformation, resulting in 95 meshes with 1001 vertices. Data2 has more local variability than Data1, and is closer to the anatomical truth.

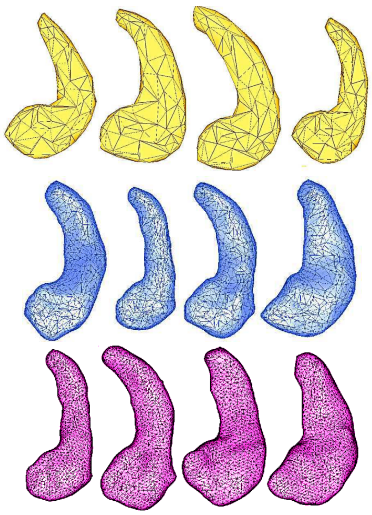

Fig. 5. Top to bottom: meshes from Data1 (n=500), Data2 (n=95) and RealData (n=95)

4.2 Effect of subject ordering

Each of the 3 proposed algorithms theoretically depends on the ordering of subjects. Here, we aim to assess the influence of the ordering of subjects on the final centroid for each algorithm.

For that purpose, we compared several centroids computed with different orderings. For each dataset and each algorithm, we computed 10 different centroids. We computed the mean $m1$ and maximal distance between all pairs of centroids. The three datasets have different variabilities. In order to relate the previous mean distance to the variability, we also computed the mean distance $m2$ between each centroid and all subjects of a given dataset. We finally computed the ratio between these two mean distances $m1/m2$. Distances between surfaces were computed in the space of currents, i.e. to compare two surfaces S and T, we computed the squared norm $\| [S] - [T] \|^2_{W^*}$. Results are presented in Table 1. Additionnaly, we computed the mean of distances between centroids computed using the different methods. Results are presented in Table 2.

Table 1. Distances between centroids computed with different subjects orderings, for each dataset and each of the 3 algorithms. The three first columns present the mean, standard deviation and the maximum of distances between all pairs of centroids computed with different orderings. The fourth colum displays the mean of distances between each centroid algorithm and all subjects of the datasets. Distances are computed in the space of currents.

		From different order:			To the dataset:	
		mean ($m1$)	max	std	mean ($m2$)	$m1/m2$
Data1	IC1	0.8682	1.3241	0.0526	91.25	0.0095
	IC2	0.5989	0.9696	0.0527	82.66	0.0072
	PW	3.5861	7.1663	0.1480	82.89	0.0433
Data2	IC1	2.4951	3.9516	0.2205	16.29	0.1531
	IC2	0.2875	0.4529	0.0164	15.95	0.0181
	PW	3.8447	5.3172	0.1919	17.61	0.2184
RealData	IC1	4.7120	6.1181	0.0944	18.54	0.2540
	IC2	0.5583	0.7867	0.0159	17.11	0.0326
	PW	5.3443	6.1334	0.1253	19.73	0.2708

Table 2. In columns, average distances between centroids computed using the different algorithms.

	IC1 vs IC2	IC1 vs PW	IC2 vs PW
Data1	1.57	5.72	6.31
Data2	1.89	3.60	3.42
RealData	3.51	5.31	4.96

For each dataset and for each type of centroid, the mean of distances between all 10 centroids is small compared to the mean of distances between the

centroid and the subjects. However, the three algorithms IC1, IC2 and PW were not equally influenced by the ordering. IC2 seems to be the most stable: the different centroids are very close one to each other, this being true for all datasets. This was expected since we reduce the matching error by combining in the space of currents the actual centroid with the deformation of the new subject along the reverse flow. For IC1, the distance was larger for Data2 and RealData, which have anatomically more realistic deformations, than for Data1, which has rather simplistic shapes. This suggests that, for real datasets, IC1 is more dependent on the ordering than IC2. This is due to the fact that IC1 provides a less precise estimate of the centroid between two shapes since it does not incorporate the reverse flow. For all datasets, distances for PW were larger than those for IC1 and IC2, suggesting that the PW algorithm is the most dependent on the subjects ordering. Furthermore, centroids computed with PW are also farther from those computed using IC1 or IC2. Furthermore, we speculate that the increased sensitivity of PW over IC1 may be due to the fact that, in IC1, $n-1$ levels of averaging are performed (and only $\log_2 n$ for PW) leading to a reduction of matching errors.

Finally, in order to provide a visualization of the differences, we present matchings between 3 centroids computed with the IC1 algorithm, in the case of RealData. Figure 6 shows that shape differences are local and residual. Visually, the 3 centroids are almost similar, and the amplitudes of momentum vectors, which bring one centroid to another, are small and local.

4.3 Position of the centroids within the population

We also assessed whether the centroids are close to the center of the population. To that purpose, we calculated the ratio

$$R = \frac{\left\| \frac{1}{N} \sum_{i=1}^{N} v_0(S_i) \right\|_V}{\frac{1}{N} \sum_{i=1}^{N} \left\| v_0(S_i) \right\|_V}, \tag{27}$$

with $v_0(S_i)$ the vector field corresponding to the initial momentum vector of the deformation from the template or the centroid to the subject i. This ratio gives some indication about the centering of the centroid, because in a pure Riemannian setting (i.e. disregarding the inaccuracies of matchings), a zero ratio would mean that we are at a critical point of the Fréchet functional, and under some reasonable assumptions on the curvature of the shape space in the neighbourhood of the dataset (which we cannot check however), it would mean that we are at the Fréchet mean. To compute R, we need to match the centroid to all subjects of the population. We computed this ratio on the best (i.e. the centroid which is the closest to all other centroids) centroid for each algorithm and for each dataset.

Results are presented in Table 3. We can observe that the centroids obtained with the three different algorithms are reasonably centered for all datasets. Centroids for Data1 are particularly well centered, which was expected given the nature of this population. Centroids for Data2 and RealData are slightly less

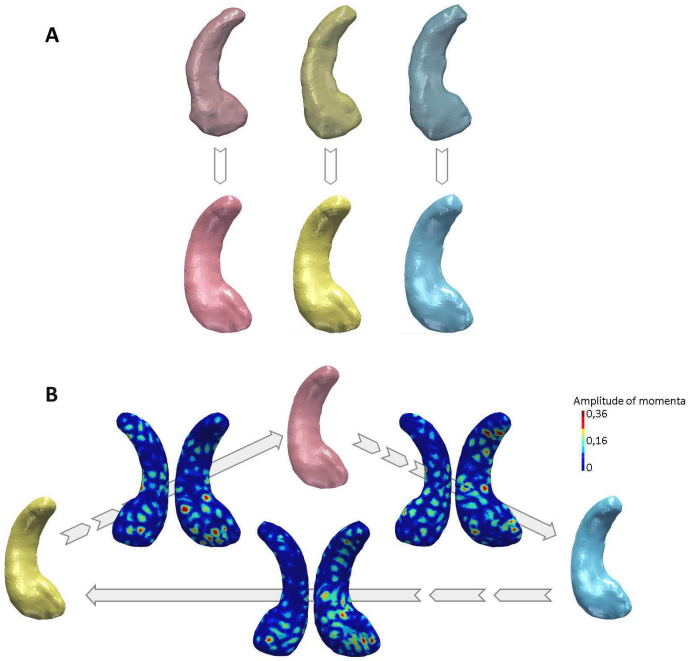

Fig. 6. A. First row: 3 initial subjects used for 3 different centroid computations with IC1 (mean distance between such centroids, in the space of currents, is 4.71) on RealData. Second row: the 3 centroids computed using the 3 subjects from the first row as initialization. B: Maps of the amplitude of the momentum vectors that map each centroid to another. Top and bottom views of the maps are displayed. One can note that the differences are small and local.

well centered but they are still close to the Fréchet mean. It is likely that using more accurate matchings (and thus increasing the computation time of the algorithms) we could reduce this ratio for RealData and Data2. Besides, one can note that ratios for Data2 and RealData are very similar; this indicates that the centering of the centroid is not altered by the variability of mesh structures in the population.

Table 3. Ratio values for assessing the position of the representative centroid within the population, computed using Equation 27 (for each algorithm and for each dataset).

R	IC1	IC2	PW
Data1	0.046	0.038	0.085
Data2	0.106	0.102	0.107
RealData	0.106	0.107	0.108

4.4 Effects of initialization on estimated template

The initial idea was to have a method which provides a good initialization for template estimation methods for large databases. We just saw that IC1 and IC2 centroids are both reasonably centered and do not depend on the subjects ordering. Despite the fact that IC2 has the smallest sensitivity to the subjects ordering, the method is slower and provides a centroid composed of N meshes. Because we want to decrease the computation time for the template estimation of a large database, it is natural to choose as initialization a centroid composed by only one mesh (time saved in kernel convolution) in a short time. We advocate to choose IC1 over PW because we can stop the IC1 algorithm at any step to get a centroid of the sub-population used so far. Furthermore, PW seems to be more sensitive to subjects ordering.

Now, we study the impact of the use of a centroid, computed with the IC1 algorithm, as initialization for the template estimation method presented in section 3.1. To that purpose, we compared the template obtained using a standard initialization, denoted as $T(StdInit)$, to the template initialized with IC1 centroid, denoted as $T(IC1)$. We chose to stop the template estimation method after 7 iterations of the optimization process. We arbitrarily chose this number of iterations, it is large enough to have a good convergence for $T(IC1)$ and to have an acceptable convergence for $T(StdInit)$. We did not use a stopping criterion based on the W^* metric because it is highly dependent on the data and is difficult to establish when using a multiscale approach. In addition to comparing $T(IC1)$ to $T(StdInit)$, we also compared the templates corresponding to two different IC1 intialization based on two different orderings. We compared the different templates in the space of currents. Results are presented in Table 4. We also computed the same ratios R as in equation 27. Results are presented in Table 5.

Table 4. Distances between templates initialized via differents IC1 ($T(IC1)$) for each datasets, and the distance between template initialized via the standard initialization ($T(StdInit)$) and templates initialized via IC1.

	$T(IC1)$ vs $T(IC1)$	$T(IC1)$ vs $T(StdInit)$
Data1	0.9833	40.9333
Data2	0.6800	20.4666
RealData	4.0433	26.8667

Table 5. Ratios R for templates ($T(IC1)$) and for the template with its usual initialization $T(StdInit)$, for each datasets.

R	$T(IC1)$	$T(StdInit)$
Data1	0.0057	0.0062
Data2	0.0073	0.0077
RealData	0.0073	0.0074

Fig. 7. Estimated template from RealData. On the left, initialized via the standard initialization which is the whole population. On the right, estimated template initialized via a IC1 centroid

One can note that the differences between $T(IC1)$ for different orderings are small for Data1 and Data2 and larger for RealData, suggesting that these are due to the mesh used for the initialization step. We can also observe that templates initialized via IC1 are far, in terms of distances in the space of currents, from the template initialized by the standard initialization. These results could be alarming, but the results of ratios (see Table 5) prove that templates are all very close to the Fréchet mean, and that the differences are not due to a bad template estimation. Moreover, both templates are visually similar as seen on Figure 7.

4.5 Effect of the number of iterations for Iterative Centroids

Since it is possible to stop the Iterative Centroid methods IC1 and IC2 at any step, we wanted to assess the influence of computing only a fraction of the N iterations on the estimated template. Indeed one may wonder if computing an I.C. at e.g. 40% (then saving 60% of computation time for the IC method) could be enough to initialize a template estimation. Moreover, for large datasets, the last subject will have a very small influence: for a database composed of 1000 subjects, the weight of the last subject is $1/1000$. We performed this experiment in the case of IC1. In the following, we call "IC1 at x%" an IC1 computed using $x \times N/100$ subjects of the population.

We computed the distance in the space of currents between "IC1 at x%" and IC1. Results are presented on Figure 8. These distances are averaged over the 10 centroids computed for each datasets. We can note that after processing 40% of the population, the IC1 covers more than 75% of the distance to the final centroid for all datasets.

We also compared T(IC1 at 40%) to T(IC1) and to T(StdInit), using distances in the space of currents, as well as the R ratio defined in Equation 27. Results are shown on Table 6). They show that using 40% of subjects lowers substantially the quality of the resulting template. Indeed the estimated template seems trapped in the local minimum found by the IC1 at 40%. We certainly have to take into account the size of the dataset. Nevertheless, we believe that if the dataset is very large and sufficiently homogeneous we could stop the Iterative Centroid method before the end.

Table 6. Results of initialization of template estimation method by a IC1 at 40%

	Data1	Data2	RealData
T(IC1 at 40%) vs $T(StdInit)$	41.41	24.41	24.82
T(IC1 at 40%) vs T(IC1 at 100%)	9.36	9.56	6.18
R value for T(IC1 at 40%)	0.040	0.106	0.105

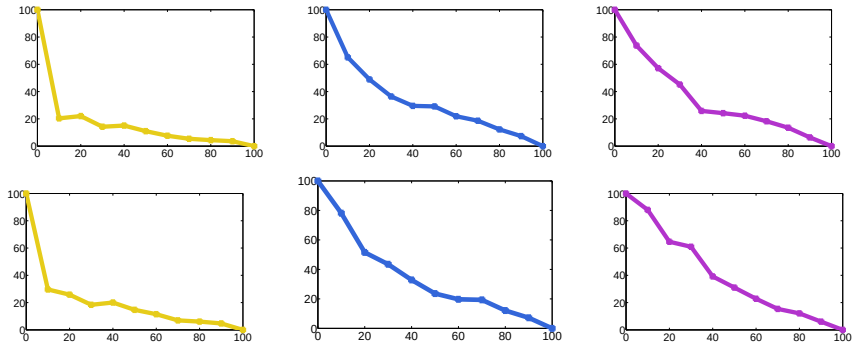

Fig. 8. first row: Graphs of average W^*-distances between the IC1 at $x\%$ and the final one. The second row present the same results with IC2.

4.6 Computation time

To speed up the matchings, we use a GPU implementation for the computation of kernel convolutions, which constitutes the most time-consuming part of LDDMM methods. Computations were performed on a Nvidia Tesla C1060 card. Computation times are displayed in Table 7.

We can note that the computation time of IC1 is equal to the one of PW and that these algorithms are faster than the IC2 algorithm, as expected. The computation time for any IC method (even for IC2) is much lower (by a factor from 10 to 80) than the computation time of the template estimation method. Morevoer, initializing the template estimation with IC1 can save up to 70% of computation time over the standard initialization. On the other hand, using T(IC1 at 40%) does not reduce computation time compared to using T(IC1).

It could be interesting to evaluate the parameters which would lead to a more precise centroid estimate in a time that would still be inferior to that needed for the template estimation. We should also mention that one could speed up computations by adding a Matching Pursuit on currents as described in [9].

5 Conclusion

We have proposed a new approach for the initialization of template estimation methods. The aim was to reduce computation time by providing a rough initial estimation, making more feasible the application of template estimation on large databases.

To that purpose, we proposed to iteratively compute a centroid which is correctly centered within the population. We proposed three different algorithms to compute this centroid: the first two algorithms are iterative (IC1 and IC2) and the third one is recursive (PW). We have evaluated the different approaches on one real and two synthetic datasets of brain anatomical structures. Overall, the centroids computed with all three approaches are close to the Fréchet

Table 7. Computation time (in hours) for Iterative Centroids and for template estimation initialised by IC1 ($T(IC1)$), the standard initialization ($T(StdInit)$) and by IC1 at 40% (T(IC1 at 40%)). For $T(IC1)$, we give the complete time for the whole process i.e. the time for the IC1 computation plus the time for $T(IC1)$ computation itself.

Computation time (hrs)	Data1	Data2	RealData
IC1	1.7	0.7	1.2
IC2	5.2	2.4	7.5
PW	1.4	0.7	1.2
$T(IC1)$	$21.1(= 1.7 + 19.4)$	$13.3(= 0.7 + 12.6)$	$27.9(= 1.2 + 26.7)$
$T(StdInit)$	96.1	20.6	99
T(IC1 at 40%)	$24.4(= 0.7 + 23.7)$	$10.4(= 0.3 + 10.1)$	$40.7(= 0.5 + 40.2)$

mean of the population, thus providing a reasonable centroid or initialization for template estimation method. Furthermore, for all methods, centroids computed using different orderings are similar. It can be noted that IC2 seems to be more robust to the ordering than IC1 which in turns seems more robust than PW. Nevertheless, in general, all methods appear relatively robust with respect to the ordering.

The advantage of iterative methods, like IC1 and IC2, is that we can stop the deformation at any step, resulting in a centroid built with part of the population. Thus, for large databases (composed for instance of 1000 subjects), it may not be necessary to include all subjects in the computation since the weight of these subjects will be very small. The iterative nature of IC1 and IC2 provides another interesting advantage which is the possible online refinement of the centroid estimation as new subjects are entered in the dataset. This leads to an increased possibility of interaction with the image analysis process. On the other hand, the recursive PW method has the advantage that it can be parallelized (still using GPU implementation), although we did not implement this specific feature in the present work.

Using the centroid as initialization of the template estimation can substantially speed up the convergence. For instance, using IC1 (which is the fastest one) as initialization saved up 70% of computation time. Moreover, this method could certainly be used to initialize other template estimation methods, such as the method proposed by Durrleman et al [6].

As we observed, the centroids, obtained with rough parameters, are close to the Fréchet mean of the population, thus we believe that computing IC with more precise parameters (but still reasonable in terms of computation time), we could obtain centroids closer to the center. This accurate centroid could be seen as a cheap alternative to true template estimation methods, particularly if computing a precise mean of the population of shapes is not required. Indeed, in the LDDMM framework, template-based shape analysis gives only a first-order, linearized approximation of the geometry in shape space. In a future work, we

will study the impact of using IC as a cheap template on results of population analysis based for instance on kernel principal component analysis. Finally, the present work deals with surfaces for which the metric based on currents seems to be well-adapted. Nevertheless, the proposed algorithms for centroid computation are general and could be applied to images, provided that an adapted metric is used.

Acknowledgments The authors are grateful to Vincent Frouin, Jean-Baptiste Poline, Roberto Toro and Edouard Duschenay for providing a sample of subjects from the IMAGEN dataset and to Marie Chupin for the use of the SACHA software.

The authors thank Professor Thomas Hotz for his suggestion on the pairwise centroid, during the discussion of the GSI'13 conference.

The research leading to these results has received funding from ANR (project HM-TC, grant number ANR-09-EMER-006, and project KaraMetria, grant number ANR-09-BLAN-0332) and from the program Investissements d´ avenir ANR-10-IAIHU-06.

References

1. Grenander, U., Miller, M.I.: Computational anatomy: An emerging discipline. Quarterly of applied mathematics **56**(4) (1998) 617–694
2. Christensen, G.E., Rabbitt, R.D., Miller, M.I.: Deformable templates using large deformation kinematics. Image Processing, IEEE Transactions on **5**(10) (1996) 1435–1447
3. Beg, M.F., Miller, M.I., Trouvé, A., Younes, L.: Computing large deformation metric mappings via geodesic flows of diffeomorphisms. International Journal of Computer Vision **61**(2) (2005) 139–157
4. Vaillant, M., Miller, M.I., Younes, L., Trouvé, A.: Statistics on diffeomorphisms via tangent space representations. NeuroImage **23** (2004) S161–S169
5. Glaunès, J., Joshi, S.: Template estimation from unlabeled point set data and surfaces for Computational Anatomy. In Pennec, X., Joshi, S., eds.: Proc. of the International Workshop on the Mathematical Foundations of Computational Anatomy (MFCA-2006). (1st of October 2006) 29–39
6. Durrleman, S., Pennec, X., Trouvé, A., Ayache, N., et al.: A forward model to build unbiased atlases from curves and surfaces. In: 2nd Medical Image Computing and Computer Assisted Intervention. Workshop on Mathematical Foundations of Computational Anatomy. (2008) 68–79
7. Durrleman, S., Prastawa, M., Korenberg, J.R., Joshi, S., Trouvé, A., Gerig, G.: Topology Preserving Atlas Construction from Shape Data without Correspondence Using Sparse Parameters. In Ayache, N., Delingette, H., Golland, P., Mori, K., eds.: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2012. Volume 7512 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2012) 223–230
8. Ma, J., Miller, M.I., Trouvé, A., Younes, L.: Bayesian template estimation in computational anatomy. NeuroImage **42**(1) (2008) 252–261

9. Durrleman, S., Pennec, X., Trouvé, A., Ayache, N.: Statistical models of sets of curves and surfaces based on currents. Medical Image Analysis **13**(5) (2009) 793–808
10. Arnaudon, M., Dombry, C., Phan, A., Yang, L.: Stochastic algorithms for computing means of probability measures. Stochastic Processes and their Applications **122**(4) (2012) 1437–1455
11. Ando, T., Li, C.K., Mathias, R.: Geometric means. Linear algebra and its applications **385** (2004) 305–334
12. Schwartz, L.: Théorie des distributions. Bull. Amer. Math. Soc. 58 (1952), 78-85 (1952) 0002–9904
13. de Rham, G.: Variétés différentiables. Formes, courants, formes harmoniques. Actualits Sci. Ind., no. 1222, Publ. Inst. Math. Univ. Nancago III., Hermann, Paris (1955)
14. Vaillant, M., Glaunes, J.: Surface matching via currents. In Christensen, G.E., Sonka, M., eds.: Information Processing in Medical Imaging. Volume 3565 of Lecture Notes in Computer Science., Springer, Springer (2005) 381–392
15. Glaunes, J.: Transport par difféomorphismes de points, de mesures et de courants pour la comparaison de formes et l'anatomie numérique. PhD thesis, Université Paris 13 (2005)
16. Durrleman, S.: Statistical models of currents for measuring the variability of anatomical curves, surfaces and their evolution. PhD thesis, University of Nice-Sophia Antipolis (2010)
17. Yang, X.F., Goh, A., Qiu, A.: Approximations of the Diffeomorphic Metric and Their Applications in Shape Learning. In: Information Processing in Medical Imaging:IPMI. (2011) 257–270
18. Tenenbaum, J., Silva, V., Langford, J.: A Global Geometric Framework for Nonlinear Dimensionality Reduction. Science **290**(5500) (December 2000) 2319–2323
19. Kendall, W.S.: Probability, convexity, and harmonic maps with small image I: uniqueness and fine existence. Proceedings of the London Mathematical Society **3**(2) (1990) 371–406
20. Karcher, H.: Riemannian center of mass and mollifier smoothing. Communications on pure and applied mathematics **30**(5) (1977) 509–541
21. Le, H.: Estimation of Riemannian barycentres. LMS J. Comput. Math **7** (2004) 193–200
22. Afsari, B.: Riemannian L ˆ{p} center of mass: Existence, uniqueness, and convexity. Proceedings of the American Mathematical Society **139**(2) (2011) 655–673
23. Afsari, B., Tron, R., Vidal, R.: On the convergence of gradient descent for finding the Riemannian center of mass. SIAM Journal on Control and Optimization **51**(3) (2013) 2230–2260
24. Emery, M., Mokobodzki, G.: Sur le barycentre d'une probabilité dans une variété. In: Séminaire de probabilités. Volume 25., Springer Berlin Heidelberg (1991) 220–233
25. Cury, C., Glaunès, J.A., Colliot, O.: Template Estimation for Large Database: A Diffeomorphic Iterative Centroid Method Using Currents. In Nielsen, F., Barbaresco, F., eds.: GSI. Volume 8085 of Lecture Notes in Computer Science., Springer (2013) 103–111
26. Chupin, M., Hammers, A., Liu, R.S.N., Colliot, O., Burdett, J., Bardinet, E., Duncan, J.S., Garnero, L., Lemieux, L.: Automatic segmentation of the hippocampus and the amygdala driven by hybrid constraints: Method and validation. NeuroImage **46**(3) (2009) 749–761

27. Glaunes, J., Trouve, A., Younes, L.: Diffeomorphic matching of distributions: a new approach for unlabelled point-sets and sub-manifolds matching. In: Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on. Volume 2. (2004) II–712–II–718 Vol.2