Deformetrica 4: an open-source software for statistical shape analysis

Alexandre Bône, Maxime Louis, Benoît Martin, Stanley Durrleman

Deformetrica 4: an open-source software for statistical shape analysis

Alexandre Bône[†12345], Maxime Louis[†12345], Benoît Martin[12345], and Stanley Durrleman[12345]

[1] Institut du Cerveau et de la Moelle épinière, ICM, F-75013, Paris, France
[2] Inserm, U 1127, F-75013, Paris, France
[3] CNRS, UMR 7225, F-75013, Paris, France
[4] Sorbonne Université, F-75013, Paris, France
[5] Inria, Aramis project-team, F-75013, Paris, France

Abstract. Deformetrica is an open-source software for the statistical analysis of images and meshes. It relies on a specific instance of the large deformation diffeomorphic metric mapping (LDDMM) framework, based on control points: local momenta vectors offer a low-dimensional and interpretable parametrization of global diffeomorphims of the 2/3D ambient space, which in turn can warp any single or collection of shapes embedded in this physical space. Deformetrica has very few requirements about the data of interest: in the particular case of meshes, the absence of point correspondence can be handled thanks to the current or varifold representations. In addition to standard computational anatomy functionalities such as shape registration or atlas estimation, a bayesian version of atlas model as well as temporal methods (geodesic regression and parallel transport) are readily available. Installation instructions, tutorials and examples can be found at `http://www.deformetrica.org`.

Keywords: statistical shape analysis · computational anatomy · large deformation diffeomorphic metric mapping · open-source software.

1 Introduction

D'Arcy Thomson first proposed the idea to compare two distinct shapes through the ambient-space deformations that transform one into the other [17]. Many years later, this insight still proves relevant, and one of its state-of-the-art avatar is the large deformation diffeomorphic metric mapping (LDDMM) [7, 15], which offers a modern and principled framework for the construction of such transformations. Deformetrica relies on a specific instance of this framework, based on control points [7]. Section 2 details this theoretical backbone of our software, along with the current and varifold representations, which allow to handle mesh without point correspondence. Section 3 reports the competitive execution times of those core operations. Section 4 describes how this computation core is leveraged to offer ready-to-use higher level models to study shape dataset.

[†]Equal contributions.

2 Theoretical background

2.1 Control-points-based LDDMM: constructing diffeomorphisms

Deformetrica offers a low-dimensional and interpretable parametrization of diffeomorphisms of the ambient space $\mathbb{R}^d$, $d \in \{2, 3\}$. Let $(q_k)_{k=1,\dots,p}$ a set of p "control" points in $\mathbb{R}^d$ and $(\mu_k)_{k=1,\dots,p}$ be a set of p "momentum" vectors of $\mathbb{R}^d$. Those paired sets define a vector "velocity" field v of the ambient space through a convolution filter:

$$v : x \in \mathbb{R}^d \to v(x) = \sum_{k=1}^{p} K(x, q_k) \cdot \mu_k \tag{1}$$

where K is typically a gaussian kernel $K(x, y) = \exp\left(-\|x - y\|^2/\sigma^2\right)$ of kernel width $\sigma > 0$. The kernel width σ will control the typical width of the generated deformation patterns. The set of vector fields v of the form (1) is a reproducible kernel Hilbert space (RKHS) V, with norm:

$$\|v\|_V^2 = \sum_{k,l=1,\dots,n} K(q_k, q_l) \cdot \mu_k^\top \mu_l. \tag{2}$$

Evolution equations are prescribed for the control point and momentum sets, called the "Hamiltonian" equations:

$$\begin{cases} \dot{q}(t) = K(q(t), q(t)) \cdot \mu(t) \\ \dot{\mu}(t) = -\frac{1}{2} \nabla_q \left\{ K\left(q(t), q(t)\right) \cdot \mu(t)^\top \mu(t) \right\} \end{cases} \tag{3}$$

These equations are integrated using an Euler or a Runge-Kutta of order 2 scheme. Is therefore obtained a time-varying velocity field $v(x, t)$ that can be computed at any time t using equation (1) with the corresponding control points $q(t)$ and momenta $\mu(t)$.

Let $x \in \mathbb{R}^d$ be any point of the ambient space. We define the transformed point $\Phi(x)$ as the value at time 1 of the function $l : [0, 1] \mapsto \mathbb{R}^d$ with initial condition $l(0) = x$ and which obeys the ordinary differential equation:

$$l'(t) = v(l(t), t). \tag{4}$$

The obtained mapping $\Phi : \mathbb{R}^d \mapsto \mathbb{R}^d$ is a diffeomorphism of the ambient space $\mathbb{R}^d$. Mathematical details are available in [19].

Overall, the obtained diffeomorphism Φ is fully parametrized by initial sets of control points q and momenta μ: we will note $\Phi = \Phi_{q,\mu}$. This simple parametrization of a large family of diffeomorphisms paves the way to the optimization of the initial control points q and momenta μ to estimate a desired transformation of the ambient space.

On a more theoretical note, for a fixed number of control points p the obtained set of diffeomorphisms has the structure of a finite-dimensional manifold, its

geodesics are defined by the Hamiltonian equations (3), its tangent space at any point is the set of velocity fields obtained by the convolution of any momenta on the corresponding control points, and its cometric is given by the kernel matrix $[K(q_k, q_l)]_{k,l=1,\dots,p}$.

2.2 Diffeomorphic action on shapes: deforming meshes or images

Once a diffeormorphism of the ambient space is constructed, the way it deforms a shape must be specified. We distinguish the cases of mesh data and image data. A diffeormophism acts on a mesh by direct and independent application onto its vertices. On an image $I : \mathbb{R}^d \mapsto \mathbb{R}$, a diffeomorphism acts according to:

$$\Phi_{q,\mu}(I) = I \circ \Phi_{q,\mu}^{-1}.$$

This computation is done the following way:

1. A initial regular grid of points $(s_k)_{k=1,\dots,r}$ corresponding to the voxel positions of the original image I is determined.
2. The positions $\Phi^{-1}(s_k)$ are computed. This is achieved using equation (4) for $k \in \{1, \dots, r\}$, integrated from 1 to 0, with initial position $l(s_k) = s_k$ and using the opposite of the momenta $\mu(t)$ describing the diffeomorphism. This operation is exactly as expensive as the computation of the deformation of a mesh with r vertices.
3. The intensities at the positions $\Phi^{-1}(s_k)$ are computed by bi/tri-linear interpolation from the original image intensities, and assigned as being the intensity of the deformed image on the grid at position s_k. Zero padding is applied outside the original image. This operation is massively parellelizable.

In the rest of the paper, we will note $\Phi_{q,\mu} \star S$ the result of the action of a diffeomorphism $\Phi_{q,\mu}$ on a shape S.

2.3 Shape attachments: evaluting deformation residuals

To evaluate if the deformed shape is close to its target, a metric is needed. For images, the Euclidian ℓ^2 distance is trivially available. For meshes, the same ℓ^2 metric can be used if there is a point-to-point correspondence. In the general case of meshes without point correspondence, the "current" or "varifold" distances are available, and described in the rest of this section.

Whether the connectivity of the mesh is made of segments or triangles, it is possible to compute the centers $(c_k)_{k=1,\dots,r}$ and the normals $(n_k)_{k=1,\dots,r}$ of the edges. Equipped with those, one can compute either the current distance [18]:

$$d\left((n_k^\alpha, c_k^\alpha)_{p=1,\dots,r^\alpha}, (n_l^\beta, c_l^\beta)_{l=1,\dots,r^\beta}\right)^2 = \sum_k \sum_l K_W(c_k^\alpha, c_l^\beta) \cdot (n_k^\alpha)^\top n_l^\beta$$

or rather the varifold distance [5], which ignores the orientation of the normals:

$$d\left((n_k^\alpha, c_k^\alpha)_{k=1,\dots,r^\alpha}, (n_l^\beta, c_l^\beta)_{l=1,\dots,r^\beta}\right)^2 = \sum_k \sum_l K_W(c_k^\alpha, c_l^\beta) \cdot \frac{\left((n_k^\alpha)^\top n_l^\beta\right)^2}{\|n_k^\alpha\| \|n_l^\beta\|}$$

where K_W is a Gaussian kernel with width σ_W.

Deformetrica offers the possibility to compute simultaneous deformations of several shapes all embedded in the same ambient space $\mathbb{R}^d$. If $O^\alpha = (S_1^\alpha, \ldots, S_{n_s}^\alpha)$ and $O^\beta = (S_1^\beta, \ldots, S_{n_s}^\beta)$ are two objects constituted of n_s homologous shapes, Deformetrica computes the squared distance via:

$$d(O^\alpha, O^\beta)^2 = \sum_{k=1}^{n_s} \frac{d(S_k^\alpha, S_k^\beta)^2}{\sigma_k^2} \tag{5}$$

which is a weighted average of the squared distances of the corresponding objects. The parameters σ_k can be used to tune the relative importances of each part of the composite "multi-object" of study.

2.4 A glimpse at optimization

Each Deformetrica model leverages those deformation and attachment mechanics to define a specific cost function, that will then be optimized either by steepest gradient descent or with the limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) method [12]. Deformetrica 4 exploits the automatic differentiation functionalities offered by the PyTorch project [16] to compute the required gradients, as suggested in [11].

3 Performances

The deformation mechanics heavily rely on convolution operations, as well as computing current or varifold attachments. Computing a convolution has a quadratic numerical complexity with the number of considered points, and is therefore a very critical operator in Deformetrica. A second constraint arise with automatic differentiation memory requirements, which are also quadratic with the input data sizes in the case of a naive implementation. Deformetrica features two ways to perform convolution, both either on CPU or GPU:

- using a naive PyTorch-based code [16], typically faster for small data sizes but unreasonably memory-greedy with larger data;
- using the dedicated PyKeops library [4] which offers a PyTorch-compatible python wrapper for memory-efficient kernel operations with their derivatives. This library is typically required to deal with real-size data.

An additional performance switch is offered by the PyTorch library: all linear algebra operations can be ported directly on GPU with a single flag. Obviously, this come at the cost of an increased GPU memory usage.

Figure 1 reports typical execution times against the data size, respectively for the attachment and deformation atomic operations. The reported times include the (automatic) computation of the gradient. This benchmark has been made

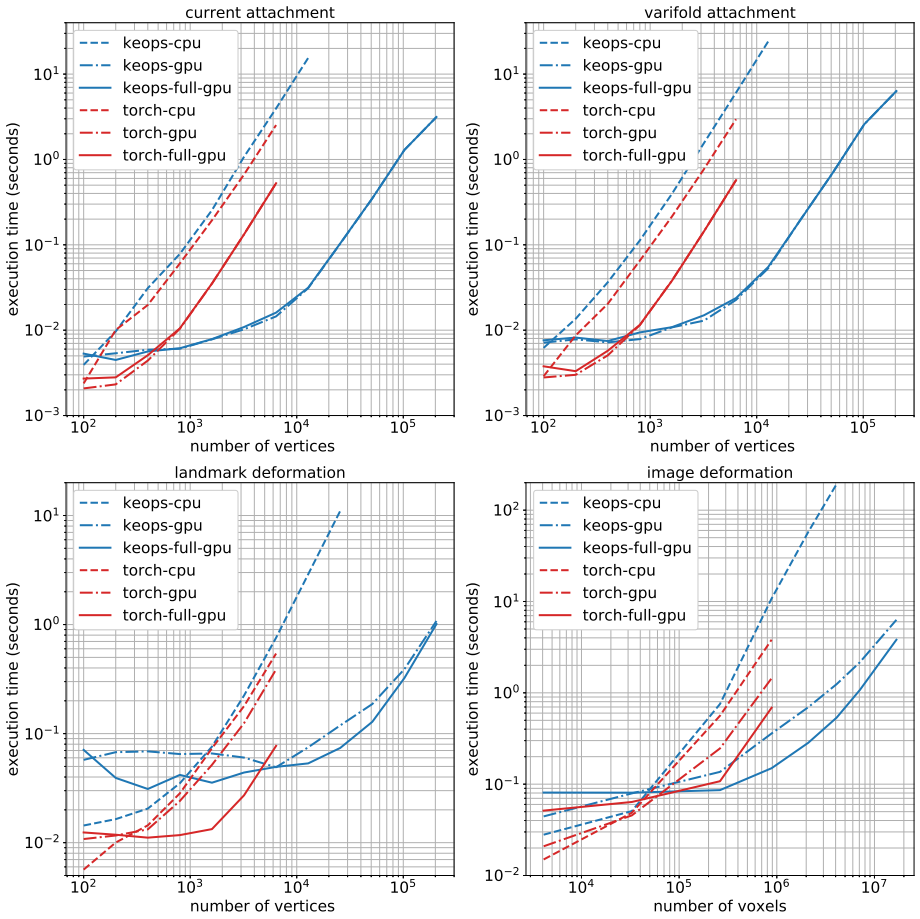

Fig. 1. Top: needed time to compute either the current or varifold attachment and the associated gradient, versus the number of vertices in each mesh. Bottom: needed time to compute either a landmark or image deformation and the associated gradient, versus the number of vertices and voxels respectively. The reported times are averages over 100 evaluations.

on an Ubuntu 14.04 machine, equipped with an Intel Xeon E5-1630 v3 CPU and Nvidia Quadro M4000 GPU with Nvidia driver version 384.130. Note that both the PyTorch and PyKeops libraries are quite recent, and can be expected to improve their performances in the near future.

In all cases, the "torch"-based convolutions are faster for small data sizes, but are overtaken by the "keops"-based ones at some point. The CPU-only operations can prove efficient to compute the deformation of small shapes, but quickly become order of magnitudes slower than their GPU equivalents for larger data. The "full-gpu" option does not lower the execution times for attachments,

when it consistently does so for deformations. Note that the torch-based curves are interrupted earlier than their keops-based counterparts, because the memory requirements due to automatic differentiation becomes unreasonable for too large data sizes.

We can finally underline the satisfyingly fast image deformation performances, allowing to register two full-resolution ($181 \times 217 \times 181$) T1-weighted magnetic resonance images (MRIs) in 1 minute and 42 seconds (after 50 iterations of the L-BFGS estimator), with a GPU memory footprint around 2 gigabytes. Choosing the slower but much less memory-intensive "keops-gpu" mode instead of "keops-full-gpu", the same registration takes 3 minutes and 22 seconds with a GPU memory footprint of 60 megabytes. In absence of gpu, the "keops-cpu" option allows to still estimate the registration, but requires around 10 hours.

4 Deformetrica applications

4.1 Atlas and registration

Cost function We consider here a cross-sectional collection of shapes $(S_i)_{i=1,\dots,n}$. The atlas model offers to compute a mean T of the shapes and a collection of diffeomorphisms $(\Phi_i)_{i=1,\dots,n}$ such that for all $i \in \{1, \dots, n\}$, we have $\Phi_i \star T \simeq S_i$. This is achieved by minimization of the cost function:

$$C(T, q, \mu_{i=1,\dots,n}) = \sum_i d(\Phi_{q,\mu_i} \star T, S_i)^2 / \sigma_\epsilon^2 + R(q, (\mu_i)_{i=1,\dots,n}), \tag{6}$$

$$\text{with} \quad R(q, (\mu_i)_{i=1,\dots,n}) = \sum_i \mu_i^\top K(q,q) \mu_i \tag{7}$$

noting $K(q,q)$ the p-by-p "kernel" matrix $[K(q_k, q_l)]_{k,l=1,\dots,p}$. The first term in equation (6) controls the data attachment i.e. how well the collection of objects is fitted by the deformation of the template, while the second term acts as a regularizer by penalizing the kinetic energy of the deformations. The relative importance of those two terms is specified by the user through the parameter σ_ϵ. The resulting atlas obtained from images of digits is displayed by Figure 4.1.

Smoothing the gradient. When working with meshes with boundaries, the gradient of the cost function (6) with respect to the mesh vertices positions T can be very large near the boundary, inducing the estimated template T to have a non-natural shape. A workaround consists in convolving the analytic gradient with a Gaussian kernel. It provides a different descent direction which results in a smoother estimated template.

Registration The registration problem is a particular instantiation of the atlas cost function with a single target S and a fixed template T:

$$C(q, \mu) = d(\Phi_{q,\mu} \star T, S)^2 / \sigma_\epsilon^2 + R(q, \mu) \tag{8}$$

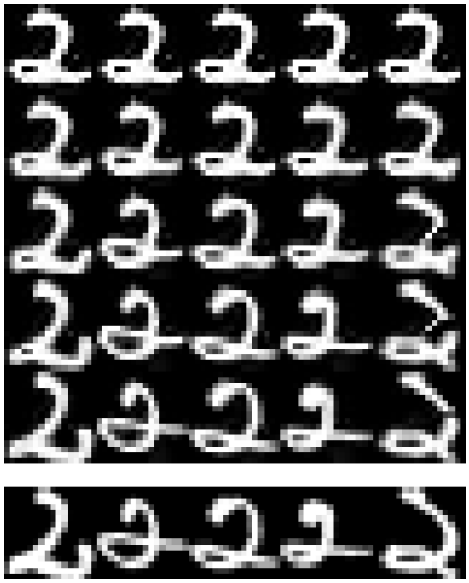

Fig. 2. Illustration of an estimated "deterministic" atlas model on the five images represented at the bottom row. The top row represents five repetitions of the estimated template shape, when the following rows represent the progressive deformations of this template that eventually match well the input dataset shown on the last row. The somehow unnatural rightmost deformation indicates that the σ_ϵ parameter might advantageously be chosen slightly greater, since less energetic deformations would be estimated.

It has numerous applications in medical imaging. For instance, registering MRIs from two different patients allows to perform relevant voxel-wise intensity comparisons, after removal of their natural anatomical differences. Alternatively, it can be leveraged to transfer some standard brain segmentation towards a new particular subject.

4.2 Bayesian Atlas

The atlas cost function (6) can be seen as an approximation of the negative complete log-likelihood of a generative, hierarchical, mixed-effects statistical model, that we call the Bayesian atlas one [10].

Statistical model From a common template T and control points q, the individual shapes S_i are considered as random deformations of T plus noise:

$$S_i = \Phi_{q,\mu_i} \star T + \epsilon_i, \quad \text{with} \quad \mu_i \overset{\text{iid}}{\sim} \mathcal{N}(0, \Sigma_\mu) \quad \text{and} \quad \epsilon_i \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma_\epsilon). \tag{9}$$

To fit the framework of mixed-effects models, we distinguish the model fixed effects $\theta = (T, q, \Sigma_\mu, \sigma_\epsilon)$ and the model random effects $z = (\mu_i)_i$. Inverse-Wishart bayesian priors are chosen for the variance parameters: $\Sigma_\mu \sim \mathcal{IW}(\Gamma_\mu, m_\mu)$ and $\sigma_\epsilon \sim \mathcal{IW}(\gamma_\epsilon, m_\epsilon)$. The introduced additional hyper-parameters are by default automatically set following the heuristics given in [10].

Log-likelihood Noting $S = (S_i)$ the collection of all the observations, the complete log-likelihood is given by:

$$-2\log p(S,\theta,z) = \sum_i \left\{ d(\Phi_{q,\mu_i} \star T, S_i)^2/\sigma_\epsilon^2 + \mu_i^\perp \Sigma_\mu^{-1} \mu_i \right\} \tag{10}$$
$$+ m_\mu \left\{ \log\left(\det \Sigma_\mu\right) + \mathrm{Tr}\left(\Sigma_\mu^\perp \Gamma_\mu^{-1}\right) \right\} + m_\epsilon \left\{ \log \sigma_\epsilon^2 + \gamma_\epsilon^2/\sigma_\epsilon^2 \right\}.$$

The maximum a posteriori (MAP) estimate of the model parameters can be approximated as follow:

$$\theta_{map} = \mathrm{argmax}_\theta \int p(S,\theta,z)\mathrm{d}z \approx \mathrm{argmax}_{\theta,z}\, p(S,\theta,z). \tag{11}$$

This classical "max-max" or "mode" approximation becomes an equality in the limit case where $p(z)$ is a Dirac distribution, i.e. $\Sigma_\mu = 0$.

Note that computing this approximate MAP amounts to finding the minimum of the negative log-likelihood (10), which echoes the previously introduced atlas cost function (6). The introduced modeling provides a statistical interpretation to the regularization term, which arises from assumed underlying random structures on the momenta μ_i and the residuals ϵ_i. Those assumptions are weaker, more intrinsic than arbitrarily prescribing the regularization term (7): the estimated atlas can therefore be expected to be more data-driven, or in other words more representative of the input data.

Estimation The Bayesian atlas is estimated in Deformetrica with gradient-based methods following the iterative procedure described in [10], which alternates gradient steps over the current estimates of $T, q, (\mu_i)$ and closed-form updates of the variance parameters $\Sigma_\mu, \sigma_\epsilon$.

A second class of estimation methods, based on a stochastic approximation of the classical expectation-maximization algorithm (see [1,6]) will be released in Deformetrica 4.1. This so-called SAEM estimator will compute the exact θ_{map}, integrating out the full distribution of the momenta random effects.

4.3 Geodesic regression

Geodesic regression generalizes linear regression to manifold-valued data [8, 9]. We consider here a time-series dataset $(S_i)_{i=1,\dots,n}$ observed at times $(t_i)_{i=1,\dots,n}$. Practical examples could be repeated MRIs of the same individual, or repeated observations of the growth of a plant. The cost function for geodesic regression is:

$$C(T,q,\mu) = \sum_i d(\Phi_{q,t_i\mu} \star T, S_i)^2/\sigma_\epsilon^2 + R(q,\mu). \tag{12}$$

where $R(q,\mu)$ is given by equation (7). The first term in (12) controls the attachment of the data while the second penalizes the "kinetic" energy of the deformation. The data-attachment versus regularity tradeoff is addressed by the

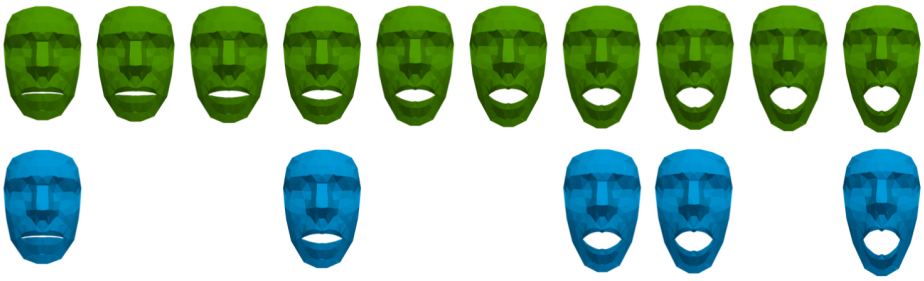

Fig. 3. Estimated geodesic regression. Top row: the estimated trajectory. Bottom row: observations from which the top trajectory is learned.

user-specified parameter σ_ϵ. Note that the trajectory $t \mapsto \Phi_{q,t\mu} \star T$ is the action of a geodesic on the q-manifold of diffeomorphisms onto the template shape T.

Optimization of this cost yields an estimated template shape T as well as sets of control points and associated initial momenta, so that the induced time-continuous flow of diffeomorphisms applied to the template shape $t \mapsto \Phi_{q,t\mu} \star T$ is as close as possible to the input observations. Figure 3 shows an example of geodesic regression on 3D meshes of human faces (data courtesy of Paolo Piras, Sapienza Universitá di Roma, Italy).

4.4 Parallel transport in shape analysis

Deformetrica implements the parallel transport method for shape analysis described in [13]. Given two sets of control points and momenta q^α, q^β and μ^α, μ^β, the parallel transport is a differential geometry notion which allows to consider the translation of the deformation described by q^β, μ^β along the deformation defined by q^α, μ^α. The computation of this transport can be done following a procedure whose convergence is proven in [14].

An interesting example occurs when q^α, μ^α describes a known progression, for example a geodesic regression learned from repeated observation of a reference subject and when q^β, μ^β describes a registration between an observation of the reference subject and a new subject. In that case, the flow of the parallel-transported deformation can be used to obtain a prediction of the future state of the subject [3]. It is in some sense a transfer learning operation.

Fig. 4. Parallel transport of the human face trajectory shown on Figure 3 onto a different face.

Figure 4 shows an example of parallel translation of the geodesic progression obtained on Figure 3 onto a face with a different form.

5 Conclusion

Deformetrica implements common computational anatomy methods both on meshes and images. Future releases of the software will include probabilistic principal geodesic analysis [20] as well as the longitudinal atlas statistical model [2].

One of the main limitation of the software for a wider range of applications lies in the purely geometrical modeling of the shapes. Mainly, a deformation model cannot change the topology of the deformed image, thus restricting the range of applications. Using the metamorphosis framework or including functional shapes could increase the impact of the software.

Acknowledgments. This work has been partly funded by the European Research Council (ERC) under grant agreement No 678304, European Union's Horizon 2020 research and innovation program under grant agreement No 666992, and the program Investissements d'avenir ANR-10-IAIHU-06.

References

1. S. Allassonnière, E. Kuhn, and A. Trouvé. Construction of bayesian deformable models via a stochastic approximation algorithm: a convergence study. *Bernoulli*, 16(3):641–678, 2010.
2. A. Bône, O. Colliot, and S. Durrleman. Learning distributions of shape trajectories from longitudinal datasets: a hierarchical model on a manifold of diffeomorphisms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9271–9280, 2018.
3. A. Bône, M. Louis, A. Routier, J. Samper, M. Bacci, B. Charlier, O. Colliot, S. Durrleman, A. D. N. Initiative, et al. Prediction of the progression of subcortical brain structures in alzheimers disease from baseline. In *Graphs in Biomedical Image Analysis, Computational Anatomy and Imaging Genetics*, pages 101–113. Springer, 2017.
4. B. Charlier, J. Feydy, J. A. Glaunès, and A. Trouvé. An efficient kernel product for automatic differentiation libraries, with applications to measure transport, 2017.
5. N. Charon and A. Trouvé. The varifold representation of nonoriented shapes for diffeomorphic registration. *SIAM Journal on Imaging Sciences*, 6(4):2547–2580, 2013.
6. B. Delyon, M. Lavielle, and E. Moulines. Convergence of a stochastic approximation version of the em algorithm. *Annals of statistics*, pages 94–128, 1999.
7. S. Durrleman, M. Prastawa, N. Charon, J. R. Korenberg, S. Joshi, G. Gerig, and A. Trouvé. Morphometry of anatomical shape complexes with dense deformations and sparse parameters. *NeuroImage*, 101:35–49, 2014.
8. J. Fishbaugh, M. Prastawa, G. Gerig, and S. Durrleman. Geodesic regression of image and shape data for improved modeling of 4D trajectories. In *ISBI 2014 - 11th International Symposium on Biomedical Imaging*, pages 385 – 388, Apr. 2014.

9. T. Fletcher. Geodesic regression on riemannian manifolds. In *Proceedings of the Third International Workshop on Mathematical Foundations of Computational Anatomy-Geometrical and Statistical Methods for Modelling Biological Shape Variability*, pages 75–86, 2011.
10. P. Gori, O. Colliot, L. Marrakchi-Kacem, Y. Worbe, C. Poupon, A. Hartmann, N. Ayache, and S. Durrleman. A Bayesian Framework for Joint Morphometry of Surface and Curve meshes in Multi-Object Complexes. *Medical Image Analysis*, 35:458–474, Jan. 2017.
11. L. Kühnel and S. Sommer. Computational anatomy in theano. In *Graphs in Biomedical Image Analysis, Computational Anatomy and Imaging Genetics*, pages 164–176. Springer, 2017.
12. D. C. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989.
13. M. Louis, A. Bône, B. Charlier, S. Durrleman, A. D. N. Initiative, et al. Parallel transport in shape analysis: a scalable numerical scheme. In *International Conference on Geometric Science of Information*, pages 29–37. Springer, 2017.
14. M. Louis, B. Charlier, P. Jusselin, P. Susovan, and S. Durrleman. A fanning scheme for the parallell transport along geodesics on riemannian manifolds. *"SIAM Journal on Numerical Analysis"*, 2018.
15. M. I. Miller, A. Trouvé, and L. Younes. Geodesic shooting for computational anatomy. *Journal of mathematical imaging and vision*, 24(2):209–228, 2006.
16. A. Paszke, S. Chintala, R. Collobert, K. Kavukcuoglu, C. Farabet, S. Bengio, I. Melvin, J. Weston, and J. Mariethoz. Pytorch: Tensors and dynamic neural networks in python with strong gpu acceleration, may 2017.
17. D. W. Thompson et al. On growth and form. *On growth and form.*, 1942.
18. M. Vaillant and J. Glaunès. Surface matching via currents. In *Biennial International Conference on Information Processing in Medical Imaging*, pages 381–392. Springer, 2005.
19. L. Younes. *Shapes and diffeomorphisms*, volume 171. Springer Science & Business Media, 2010.
20. M. Zhang, N. Singh, and P. T. Fletcher. Bayesian estimation of regularization and atlas building in diffeomorphic image registration. In *IPMI*, volume 23, pages 37–48, 2013.