1. Diffeomorphic locally weighted translations

Given a smooth (symmetric positive definite) kernel function $k_\rho(\mathbf{x}, \mathbf{y})$, depending on some parameter ρ, such that $\forall \mathbf{x}, k_\rho(\mathbf{x}, \mathbf{x}) = 1$ and $k_\rho(\mathbf{x}, \mathbf{y}) \to 0$ when $\|\mathbf{y} - \mathbf{x}\| \to \infty$, given a "direction" $\mathbf{v} \in \mathbb{R}^d$ and a "center" $\mathbf{c} \in \mathbb{R}^d$, we consider the following *locally weighted translation*:

$$\psi_{\rho,\mathbf{c},\mathbf{v}}(\mathbf{x}) = \mathbf{x} + k_\rho(\mathbf{x}, \mathbf{c})\mathbf{v}.$$

Theorem 1. *If* $\forall (\mathbf{x}, \mathbf{y}) \in \mathbb{R}^d \times \mathbb{R}^d$, $\frac{\partial k_\rho}{\partial \mathbf{x}}(\mathbf{x}, \mathbf{y}) \cdot \mathbf{v} > -1$, *then* $\psi_{\rho,\mathbf{c},\mathbf{v}}$ *is a diffeomorphism.*

Proof. For $\mathbf{x} \in \mathbb{R}^d$, let us define:

$$h_{\mathbf{x}} : r \in \mathbb{R} \mapsto r + k_\rho(\mathbf{x} + r\mathbf{v}, \mathbf{c}) \in \mathbb{R}.$$

If $\frac{\partial k_\rho}{\partial \mathbf{x}}(\mathbf{x}, \mathbf{y}) \cdot \mathbf{v} > -1$, we get: $\forall r \in \mathbb{R}$, $\frac{dh_{\mathbf{x}}}{dr}(r) > 0$. Since $h_{\mathbf{x}}(r)$ tends to $-\infty$ when r tends to $-\infty$, and to $+\infty$ when r tends to $+\infty$, we deduce that there exists a unique $s(\mathbf{x}) \in \mathbb{R}$ such that $h_{\mathbf{x}}(s(\mathbf{x})) = 0$. Moreover, $\psi_{\rho,\mathbf{c},\mathbf{v}}(\mathbf{x} + r\mathbf{v}) = \mathbf{x} + h_{\mathbf{x}}(r)\mathbf{v}$, thus $\psi_{\rho,\mathbf{c},\mathbf{v}}(\mathbf{x} + r\mathbf{v}) = \mathbf{x}$ if and only if $r = s(\mathbf{x})$ or $\mathbf{v} = \mathbf{0}$. Besides, any $\mathbf{y} \in \mathbb{R}^d$ verifying $\psi_{\rho,\mathbf{c},\mathbf{v}}(\mathbf{y}) = \mathbf{x}$ is necessarily of the form $\mathbf{x} + r\mathbf{v}$, so $\psi_{\rho,\mathbf{c},\mathbf{v}}(\mathbf{y}) = \mathbf{x}$ implies $\mathbf{y} = \mathbf{x} + s(\mathbf{x})\mathbf{v}$.

We conclude that $\psi_{\rho,\mathbf{c},\mathbf{v}}$ is invertible, and:

$$\psi^{-1}_{\rho,\mathbf{c},\mathbf{v}}(\mathbf{x}) = \mathbf{x} + s(\mathbf{x})\mathbf{v}.$$

Finally, the inverse function theorem can be applied to prove that $\psi_{\rho,\mathbf{c},\mathbf{v}}$ is a diffeomorphism. □

With Gaussian Radial Basis Function (RBF) kernel:

We now consider the following kernel (with $\rho \in \mathbb{R}_{>0}$):

$$k_\rho(\mathbf{x}, \mathbf{y}) = \exp\left(-\rho^2 \|\mathbf{x} - \mathbf{y}\|^2\right).$$

We have:

$$\frac{\partial k_\rho}{\partial \mathbf{x}}(\mathbf{x}, \mathbf{y}) \cdot \mathbf{v} = -2\rho^2 \exp\left(-\rho^2 \|\mathbf{x} - \mathbf{y}\|^2\right)(\mathbf{x} - \mathbf{y}) \cdot \mathbf{v},$$

with the lower bound:

$$\frac{\partial k_\rho}{\partial \mathbf{x}}(\mathbf{x}, \mathbf{y}) \cdot \mathbf{v} \geq -2\rho^2 \exp\left(-\rho^2 \|\mathbf{x} - \mathbf{y}\|^2\right) \|\mathbf{x} - \mathbf{y}\|.\|\mathbf{v}\|.$$

The expression on the right takes its minimum for $\|\mathbf{x}-\mathbf{y}\| = \frac{1}{\sqrt{2}\rho}$, which yields:

$$\frac{\partial k_\rho}{\partial \mathbf{x}}(\mathbf{x}, \mathbf{y}) \cdot \mathbf{v} \geq -\sqrt{2}\|\mathbf{v}\|\rho \exp\left(-\frac{1}{2}\right).$$

Applying Theorem 1, $\mathbf{v} = 0$ or $\rho < \rho_{\max}(\mathbf{v}) = \frac{1}{\sqrt{2}\|\mathbf{v}\|} \exp\left(\frac{1}{2}\right)$ implies that $\psi_{\rho,\mathbf{c},\mathbf{v}}$ is a diffeomorphism.

In that case, $s(\mathbf{x})$, and as a result $\psi^{-1}_{\rho,\mathbf{c},\mathbf{v}}(\mathbf{x})$, can be very efficiently computed with Newton's method.

2. A diffeomorphic matching algorithm

In this section we are interested in the following problem: given two sequences of distinct points $\mathbf{X} = (\mathbf{x}_i)_{i\in\{0,\dots,N\}}$ and $\mathbf{Y} = (\mathbf{y}_i)_{i\in\{0,\dots,N\}}$, compute a diffeomorphism Φ that maps each $\mathbf{x}_i$ onto $\mathbf{y}_i$, either exactly or approximately. More formally, defining $\mathsf{dist}(\mathbf{A}, \mathbf{B}) = \frac{1}{N+1}\sum_i \|\mathbf{a}_i - \mathbf{b}_i\|^2$ for two sequences $\mathbf{A}$ and $\mathbf{B}$ of $N+1$ points, and denoting by $\Phi(\mathbf{X})$ the sequence of points $(\Phi(\mathbf{x}_i))_{i\in\{0,\dots,N\}}$, we want to find a diffeomorphism Φ that minimizes $\mathsf{dist}(\Phi(\mathbf{X}), \mathbf{Y})$.

Since the sequences $\mathbf{X}$ and $\mathbf{Y}$ can be very different in shape, to the best of our knowledge the state-of-the-art existing techniques to solve this problem are based on the Large Deformation Diffeomorphic Metric Mapping (LDDMM) framework introduced in the seminal article by Joshi and Miller [1]. Its core idea is to work with a time dependent vector field $v(\mathbf{x}, t) \in \mathbb{R}^d$ ($t \in [0, 1]$), and define a flow $\phi(\mathbf{x}, t)$ via the transport equation:

$$\frac{d\phi(\mathbf{x}, t)}{dt} = v(\phi(\mathbf{x}, t), t),$$

with $\phi(\mathbf{x}, 0) = \mathbf{x}$. With a few regularity conditions on v (see [2] for specific requirements), $\mathbf{x} \mapsto \phi(\mathbf{x}, t)$ is a diffeomorphism. The resulting diffeomorphism $\Phi(\mathbf{x}) = \phi(\mathbf{x}, 1)$ is given by:

$$\Phi(\mathbf{x}) = \mathbf{x} + \int_0^1 v(\phi(\mathbf{x}, t), t)dt.$$

Using an appropriate Hilbert space for the vector fields $\mathbf{x} \mapsto v(\mathbf{x}, t)$, they can be associated with an infinitesimal cost whose integration is interpreted as a deformation energy.

Various gradient descent algorithms have been proposed to optimize v with respect to a cost that depends both on the deformation energy and on the accuracy of the mapping, whether the objective is to map curves [3], surfaces [4], or, as in our case, points [5].

The LDDMM framework has several advantages. For example, it tries to minimize the deformation, and allows the computation of similarity measures between diffeomorphic geometrical objects. However, Φ is not in closed-form, so once obtained, evaluating it requires an integration that can be slightly time-costly. In our context, Φ can be used inside a control law, so its evaluation (and that of Φ^{-1}) needs to be very fast.

We propose a completely different approach to diffeomorphic matching, based on the diffeomorphic locally weighted translations presented in the previous section, which are functions that can be evaluated extremely quickly. The proposed algorithm is based on simple principles, yet it does not seem to have been studied in the past.

2.1. The algorithm

We fix a number of iterations K, and two parameters $0 < \mu < 1$ and $0 < \beta \leq 1$. Typically, on the examples presented in this paper, we use $K = 150$, $\mu \approx 0.9$ and $\beta \approx 0.5$.

Initially, we define $\mathbf{Z} := \mathbf{X}$. Every iteration j updates $\mathbf{Z}$, and can be briefly described by the three following steps:

1. we select the point $\mathbf{p}_j$ in $\mathbf{Z}$ that is the furthest from its corresponding target $\mathbf{q}$ in $\mathbf{Y}$;

2. we consider the locally weighted translation $\psi_{\rho_j,\mathbf{p}_j,\mathbf{v}_j}$ of direction $\mathbf{v}_j = \beta(\mathbf{q}-\mathbf{p}_j)$, center $\mathbf{p}_j$, and Gaussian RBF kernel k_{ρ_j}, optimizing $\rho_j \in [0, \mu\rho_{\max}(\mathbf{v}_j)]$ to minimize the error between $\psi_{\rho_j,\mathbf{p}_j,\mathbf{v}_j}(\mathbf{Z})$ and $\mathbf{Y}$;
3. we perform the update: $\mathbf{Z} := \psi_{\rho_j,\mathbf{p}_j,\mathbf{v}_j}(\mathbf{Z})$.

The resulting diffeomorphism is:

$$\Phi = \psi_{\rho_K,\mathbf{p}_K,\mathbf{v}_K} \circ \cdots \circ \psi_{\rho_2,\mathbf{p}_2,\mathbf{v}_2} \circ \psi_{\rho_1,\mathbf{p}_1,\mathbf{v}_1}$$

Here is a description of the algorithm in pseudo-code:

1: **Input:** $\mathbf{X} = (\mathbf{x}_i)_{i\in\{0,\dots,N\}}$ and $\mathbf{Y} = (\mathbf{y}_i)_{i\in\{0,\dots,N\}}$
2: **Parameters:** $K \in \mathbb{N}_{>0}$, $0 < \mu < 1$, $0 < \beta \leq 1$
3:
4: $\mathbf{Z} = (\mathbf{z}_i)_{i\in\{0,\dots,N\}}$
5: $\mathbf{Z} := \mathbf{X}$
6: **for** $j = 1$ to K **do**
7: $\quad m := \underset{i\in\{0,\dots,N\}}{\arg\max} (\|\mathbf{z}_i - \mathbf{y}_i\|)$
8: $\quad \mathbf{p}_j := \mathbf{z}_m$
9: $\quad \mathbf{q} := \mathbf{y}_m$
10: $\quad \mathbf{v}_j := \beta(\mathbf{q} - \mathbf{p}_j)$
11: $\quad \rho_j := \underset{\rho\in[0,\mu\rho_{\max}(\mathbf{v}_j)]}{\arg\min} (\mathsf{dist}(\psi_{\rho,\mathbf{p}_j,\mathbf{v}_j}(\mathbf{Z}), \mathbf{Y}))$
12: $\quad \mathbf{Z} := \psi_{\rho_j,\mathbf{p}_j,\mathbf{v}_j}(\mathbf{Z})$
13: **end for**
14: **return** $(\rho_j)_{j\in\{1,\dots,K\}}$, $(\mathbf{p}_j)_{j\in\{1,\dots,K\}}$, $(\mathbf{v}_j)_{j\in\{1,\dots,K\}}$

Remarks:

- Here we have presented a version of the algorithm in which the parameter β is constant, but we can also make it vary iteration after iteration, for example by slowly increasing towards 1.
- The line 11 of the algorithm performs a nonlinear optimization, but since it depends only on one bounded real variable, a minimum can be found very quickly and precisely.
- We can add a fixed upper bound $\rho_M > 0$ for all ρ_j, and a regularization term in the cost of the optimization problem of line 11, to prevent the diffeomorphism from overly deforming the space to get a perfect matching. Simply using sequences with a large number of points has a similar effect (and it almost does not slow the algorithm down).
- Again in line 11, dist can be replaced by any other distance. In practice, writing $\mathbf{X}$ and $\mathbf{Y}$ as $(N+1)$-by-2 matrices and taking the largest singular value norm of $(\mathbf{X}-\mathbf{Y})$ seems to give slightly better results than with the Euclidean distance.
- Nothing prevents the algorithm from getting stuck in a local minimum, so a general proof of convergence cannot be found. However, as we show in the next sections, experimental results give empirical evidence that the algorithm is efficient and converges quickly in practice, even on difficult matching problems. In future work, we will try to further improve the algorithm and get convergence proofs under realistic assumptions.

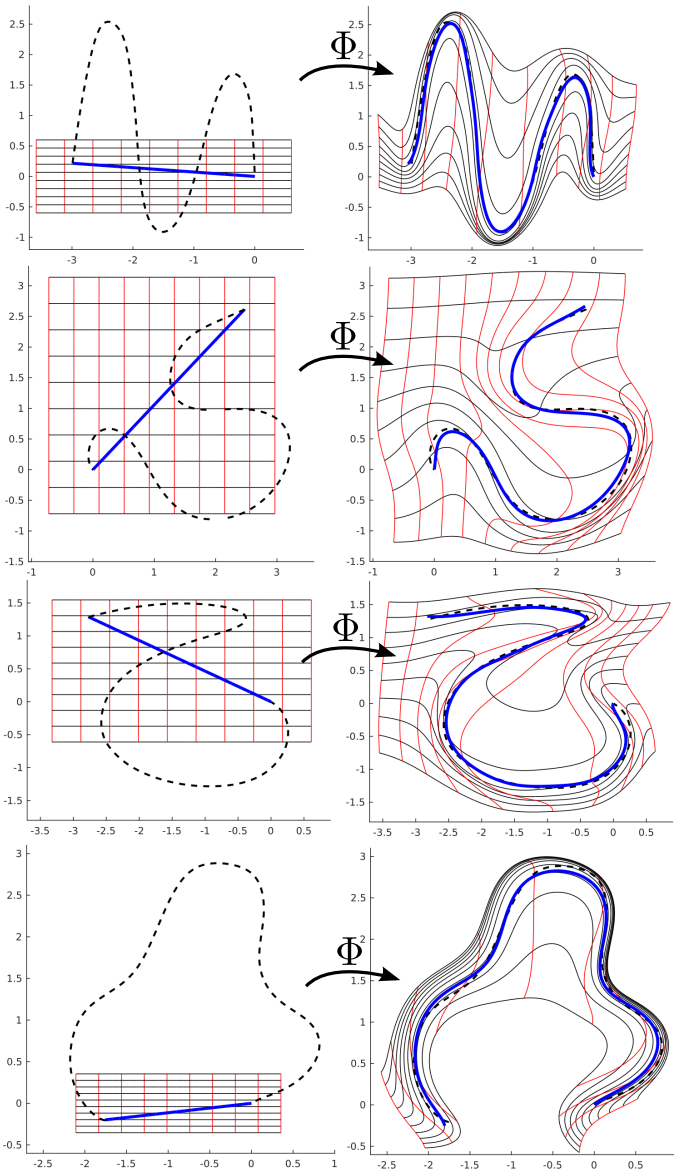

Figure 1: On the left, the dashed curve is a trajectory represented by a sequence of points $\mathbf{Y} = (\mathbf{y}_i)_{i\in\{0,\dots,N\}}$. The solid line is $\mathbf{X} = (\mathbf{y}_0 + \frac{i}{N}(\mathbf{y}_N - \mathbf{y}_0))_{i\in\{0,\dots,N\}}$. The right side shows the result of the application of the diffeomorphism Φ constructed by our algorithm to map $\mathbf{X}$ onto $\mathbf{Y}$.

2.2. *Experimental evaluation*

We compare our algorithm to an implementation of diffeomorphic matching in the LDDMM framework developed by J. Glaunès (the "`Matchine`" software [6]).

Given a sequence of points $\mathbf{Y} = (\mathbf{y}_i)_{i\in\{0,\dots,N\}}$ representing a trajectory, we set $\mathbf{X} = (\mathbf{x}_i)_{i\in\{0,\dots,N\}} = \left(\mathbf{y}_0 + \frac{i}{N}(\mathbf{y}_N - \mathbf{y}_0)\right)_{i\in\{0,\dots,N\}}$ and apply our algorithm or the LDDMM algorithm to construct a diffeomorphism Φ such that $\Phi(\mathbf{X})$ and $\mathbf{Y}$ match. Figure 1 displays the result of our algorithm on four 2D trajectories, and Table 1 shows a comparison of the results obtained on these trajectories with our algorithm and the LDDMM algorithm. For each trajectory, we try with representations as sequences of 21, 51 and 101 points (i.e. $N = 20$, $N = 50$, $N = 100$). For both algorithms, the same parameters are kept across all the trials.

	N	our algorithm	LDDMM
Learning: average duration of the construction of Φ	20	0.25 s	2.78 s
	50	0.25 s	14.5 s
	100	0.26 s	53.3 s
Forward evaluation: average duration of the computation of $\Phi(\mathbf{X})$	20	3.05 ms	157 ms
	50	3.35 ms	804 ms
	100	3.72 ms	3130 ms
Backward evaluation: average duration of the computation of $\Phi^{-1}(\mathbf{Y})$	20	29.8 ms	145 ms
	50	35 ms	798 ms
	100	38.5 ms	3110 ms
Accuracy: average value of $\mathsf{dist}(\Phi(\mathbf{X}), \mathbf{Y})$	20	3.49×10^{-3}	18.2×10^{-3}
	50	8.32×10^{-3}	22.2×10^{-3}
	100	9.51×10^{-3}	22×10^{-3}

Table 1: Comparison of experimental results

In all cases, our algorithm provides a substantial speedup. For example, with 51 points, Φ is learned in average 58 times faster and evaluated 240 times faster, while the error $\mathsf{dist}(\Phi(\mathbf{X}), \mathbf{Y})$ is 2.67 times smaller. The tests were made on an Intel(R) Core(TM) i7-4700MQ @ 2.4 GHz with 4GB of RAM.

References

[1] S. C. Joshi, M. Miller, et al., Landmark matching via large deformation diffeomorphisms, IEEE Transactions on Image Processing 9 (8) (2000) 1357–1370.

[2] P. Dupuis, U. Grenander, M. I. Miller, Variational problems on flows of diffeomorphisms for image matching, Quarterly of applied mathematics 56 (3) (1998) 587–600.

[3] J. Glaunès, A. Qiu, M. I. Miller, L. Younes, Large deformation diffeomorphic metric curve mapping., Int J Comput Vis 80 (3) (2008) 317–336.

[4] M. Vaillant, J. Glaunès, Surface matching via currents, in: Information Processing in Medical Imaging, 2005, pp. 381–392.

[5] H. Guo, A. Rangarajan, S. Joshi, Diffeomorphic point matching, in: Handbook of Mathematical Models in Computer Vision, Springer, 2006, pp. 205–219.

[6] "Matchine" software by J. A. Glaunès, Copyright (C) Université Paris Descartes, http://www.mi.parisdescartes.fr/~glaunes/matchine.zip.