Sorting points in dimension $D > 1$

Jean Feydy
Imperial College London

Sea Ice Modeling and Data Assimilation, Dartmouth, online — April 2021.

Joint work with B. Charlier, J. Glaunès (numerical foundations),
T. Séjourné, F.-X. Vialard, G. Peyré (optimal transport theory),
P. Roussillon, P. Gori, A. Trouvé (applications to computational anatomy),
F. Sverrisson, B. E. Correia, M. Bronstein (applications to protein sciences).

Who am I?

2012–2016 ENS Paris, **mathematics** and applications.

2015 MVA thesis with **Siemens Healthcare** in Princeton.

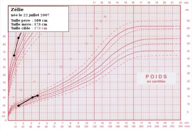
2016–2019 PhD thesis with Alain Trouvé, **computational anatomy**;
TA/tutor in applied maths at the ENS Paris.

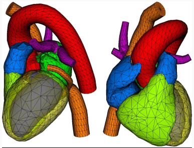
2019–2022 PostDoc with Michael Bronstein, **geometric deep learning**.

Family of medical doctors (radiologist, haematologist, GPs...):
strong motivation to work towards **clinical solutions**.

Make life easier for engineers and researchers in the field:
two libraries (KeOps, GeomLoss) to **speed up geometric methods**,
with new guarantees of **robustness**.

The medical imaging pipeline [Ptr19, EPW^{+}11]





Valuable information

Statistics

High-level description

Computational anatomy

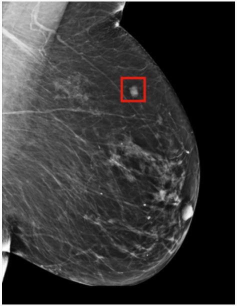
Raw image

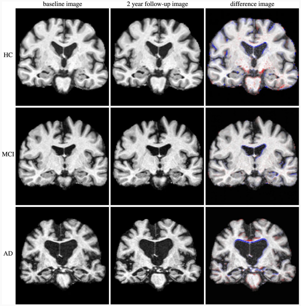
Signal processing

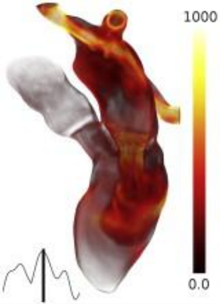
Sensor data

Computational anatomy [CSG19, LSG+18, CMN14]

Three main problems:



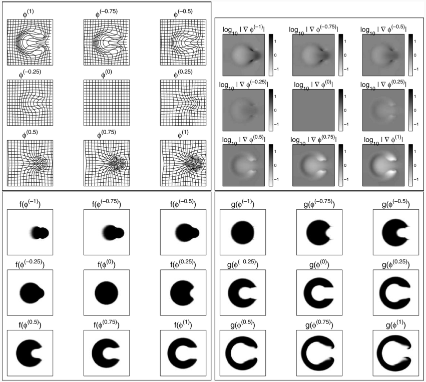




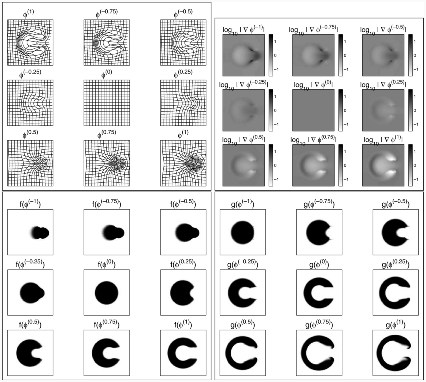
Spot patterns

Analyze variations

Fit models



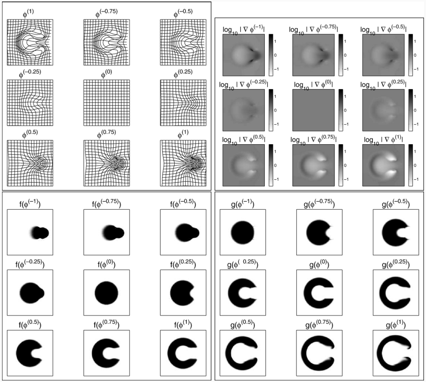
Advection for images and volumes



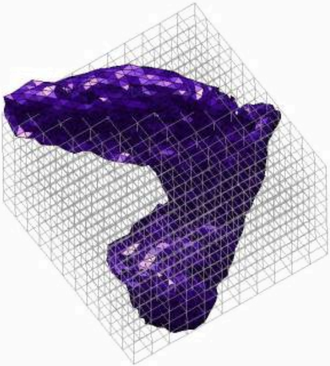
Advection for images and volumes

Mesh deformation

Shape analysis [Ash07, Gla05]

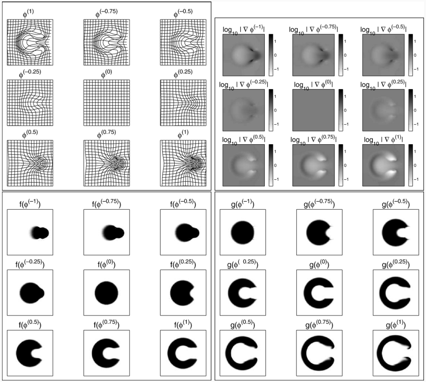


Advection for images and volumes

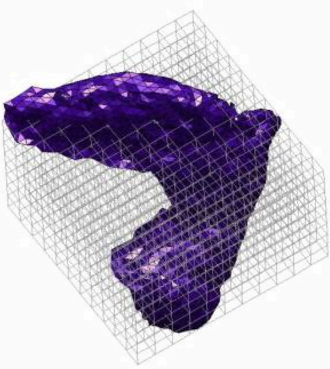


Mesh deformation

Shape analysis [Ash07, Gla05]

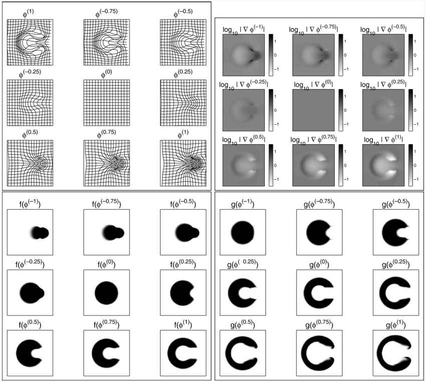


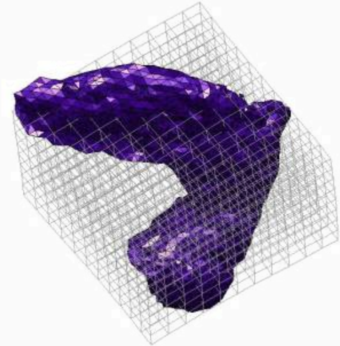
Advection for images and volumes



Mesh deformation

Shape analysis [Ash07, Gla05]

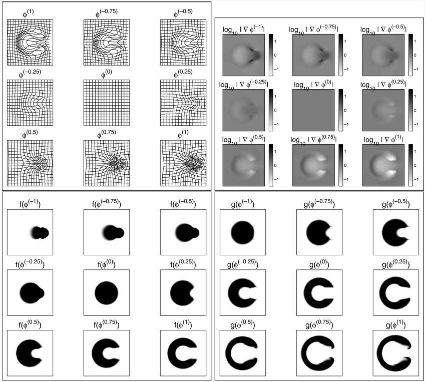


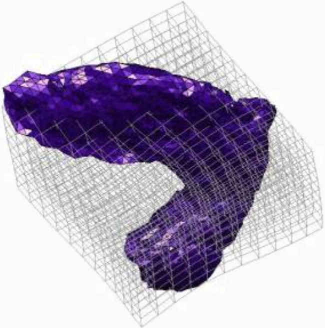


Advection for images and volumes

Mesh deformation

Shape analysis [Ash07, Gla05]

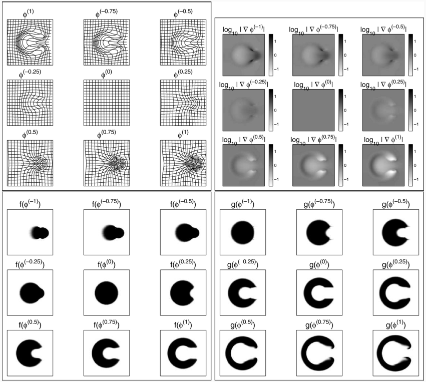




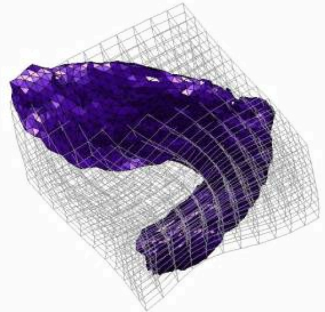
Advection for images and volumes

Mesh deformation

Shape analysis [Ash07, Gla05]

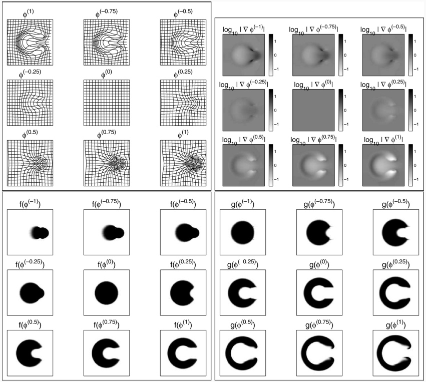


Advection for images and volumes

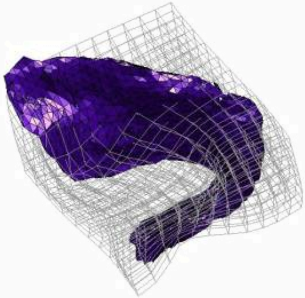


Mesh deformation

Shape analysis [Ash07, Gla05]

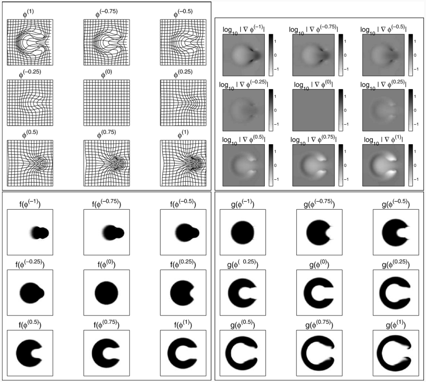


Advection for images and volumes

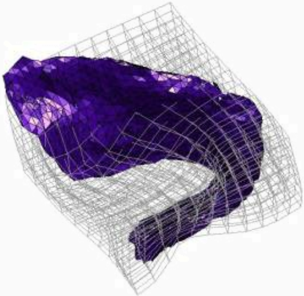


Mesh deformation

Shape analysis [Ash07, Gla05]



Advection for images and volumes



Mesh deformation

$\Longrightarrow$ We need **fast geometric primitives**.

Problem: not supported well by NumPy, TensorFlow and PyTorch.

My work so far:

- **Efficient** GPU routines for point clouds, kernels, etc.:
  - $\longrightarrow$ KeOps extension for PyTorch, NumPy, Matlab, R.
  - $\longrightarrow$ Efficient support for "symbolic" arrays.

- **Robust** deformation and feature extraction architectures:
  - $\longrightarrow$ Diffeomorphisms, elastic meshes, etc.
  - $\longrightarrow$ Geometric deep learning on protein surfaces.

- **Geometric** distances between shapes and distributions:
  - $\longrightarrow$ Wasserstein distance = optimal transport = **sorting**.
  - $\longrightarrow$ Our focus today.

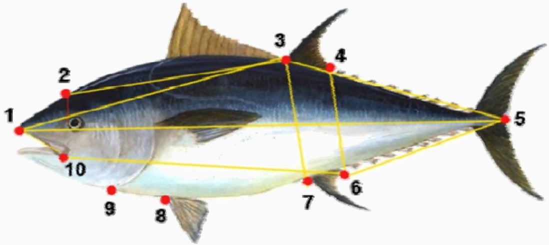
Outline of the presentation – focus on Optimal Transport (OT)

Today, we will talk about:

1. **Optimal Transport**, from a geometric perspective.
2. Modern "**QuickSort**-like" solvers on CPU and GPU.
3. Main **weaknesses** of OT tools – with some workarounds.
4. The **software suite** that is currently being built to bring reference implementations to the wider scientific community.

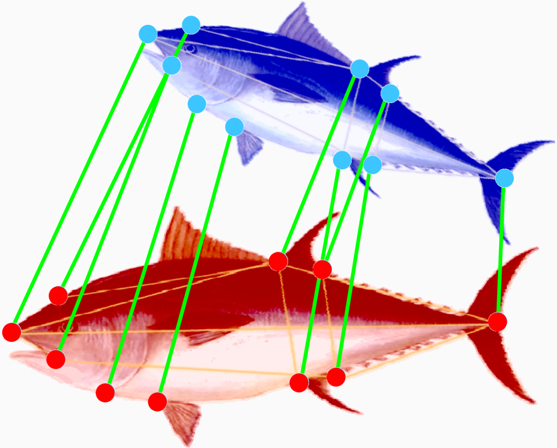
Working with unlabeled point clouds

Life is easy when you have labels



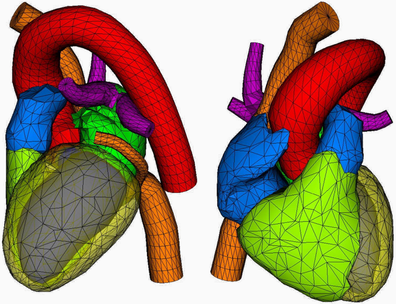
Anatomical landmarks from *A morphometric approach for the analysis of body shape in bluefin tuna*, Addis et al., 2009.

Life is easy when you have labels

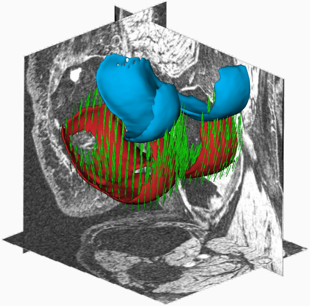


Anatomical landmarks from *A morphometric approach for the analysis of body shape in bluefin tuna*, Addis et al., 2009.

Unfortunately, medical data is often unlabeled [EPW+11]

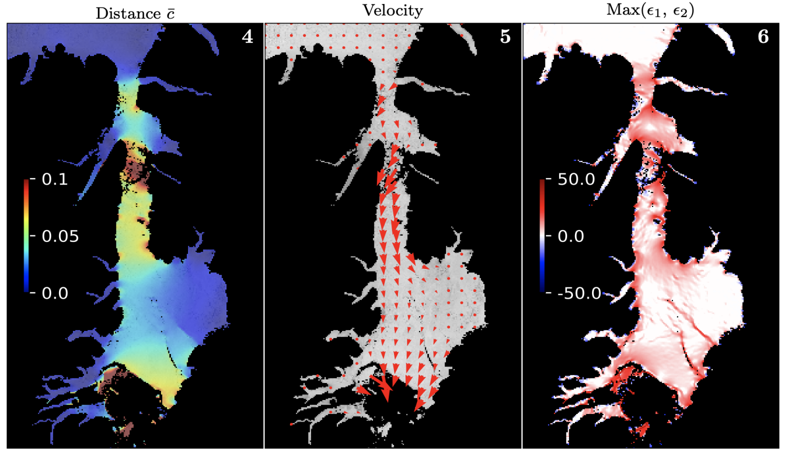


Surface meshes



Segmentation masks

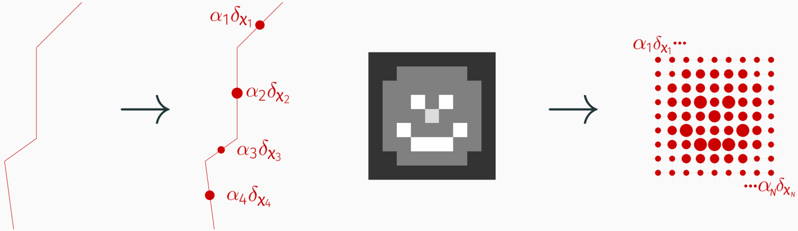
I understand that you have the same problem :-)



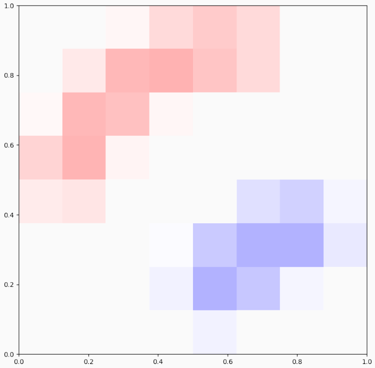
Encoding unlabeled shapes as measures

Let's enforce sampling invariance:

$$A \longrightarrow \alpha = \sum_{i=1}^{N} \alpha_i \delta_{x_i}, \qquad B \longrightarrow \beta = \sum_{j=1}^{M} \beta_j \delta_{y_j}.$$



A baseline setting: density registration



$$\alpha = \sum_{i=1}^{N} \alpha_i \delta_{x_i}, \quad \beta = \sum_{j=1}^{M} \beta_j \delta_{y_j}.$$

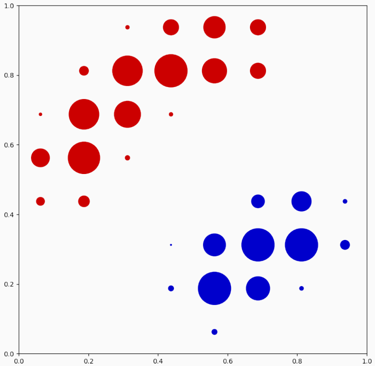
$$\sum_{i=1}^{N} \alpha_i = 1 = \sum_{j=1}^{M} \beta_j$$

Display $v_i = -\frac{1}{\alpha_i} \nabla_{x_i} \text{Loss}(\alpha, \beta)$.

Seamless extensions to:

- $\sum_i \alpha_i \neq \sum_j \beta_j$, outliers [SFV+19],
- curves and surfaces, more complex features [KCC17],
- variable weights α_i.

A baseline setting: density registration



$$\alpha = \sum_{i=1}^{N} \alpha_i \delta_{x_i}\,, \quad \beta = \sum_{j=1}^{M} \beta_j \delta_{y_j}\,.$$

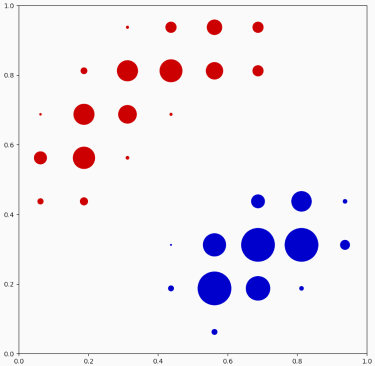
$$\sum_{i=1}^{N} \alpha_i \;=\; 1 \;=\; \sum_{j=1}^{M} \beta_j$$

Display $v_i \;=\; -\frac{1}{\alpha_i} \nabla_{x_i} \text{Loss}(\alpha, \beta)$.

Seamless extensions to:

- $\sum_i \alpha_i \neq \sum_j \beta_j$, outliers [SFV+19],
- curves and surfaces, more complex features [KCC17],
- variable weights α_i.

A baseline setting: density registration



$$\alpha = \sum_{i=1}^{N} \alpha_i \delta_{x_i}\,, \quad \beta = \sum_{j=1}^{M} \beta_j \delta_{y_j}\,.$$

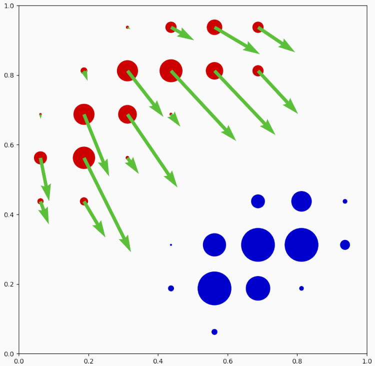
$$\sum_{i=1}^{N} \alpha_i \;=\; 1 \;=\; \sum_{j=1}^{M} \beta_j$$

Display $v_i \;=\; -\frac{1}{\alpha_i} \nabla_{x_i} \text{Loss}(\alpha, \beta)$.

Seamless extensions to:

- $\sum_i \alpha_i \neq \sum_j \beta_j$, outliers [SFV+19],
- curves and surfaces, more complex features [KCC17],
- variable weights α_i.

A baseline setting: density registration



$$\alpha = \sum_{i=1}^{N} \alpha_i \delta_{x_i}\,, \quad \beta = \sum_{j=1}^{M} \beta_j \delta_{y_j}\,.$$

$$\sum_{i=1}^{N} \alpha_i \;=\; 1 \;=\; \sum_{j=1}^{M} \beta_j$$

Display $v_i \;=\; -\frac{1}{\alpha_i} \nabla_{x_i} \mathrm{Loss}(\alpha, \beta)$.

Seamless extensions to:

- $\sum_i \alpha_i \neq \sum_j \beta_j$, outliers [SFV+19],
- curves and surfaces, more complex features [KCC17],
- variable weights α_i.

Simple distance-like functions between measures

- Nearest neighbours $\simeq$ **Chamfer** distance $\simeq$ soft-**Hausdorff**:
  Projection-based $\longrightarrow$ Degenerate gradients.

Simple distance-like functions between measures

- Nearest neighbours $\simeq$ **Chamfer** distance $\simeq$ soft-**Hausdorff**:
  Projection-based $\longrightarrow$ Degenerate gradients.

- Kernel distance $\simeq$ Blurred L^2 norm, convolution-based:

  $$\text{Loss}(\alpha, \beta) \;=\; \tfrac{1}{2}\|g \star (\alpha - \beta)\|^2_{L^2(\mathbb{R}^D)} \;=\; \tfrac{1}{2}\langle\, \alpha - \beta\,,\, k \star (\alpha - \beta)\,\rangle$$

  where $k \;=\; (g \circ (x \mapsto -x)) \star g\,.$

Simple distance-like functions between measures

- Nearest neighbours $\simeq$ **Chamfer** distance $\simeq$ soft-**Hausdorff**:
  Projection-based $\longrightarrow$ Degenerate gradients.

- Kernel distance $\simeq$ Blurred L^2 norm, convolution-based:

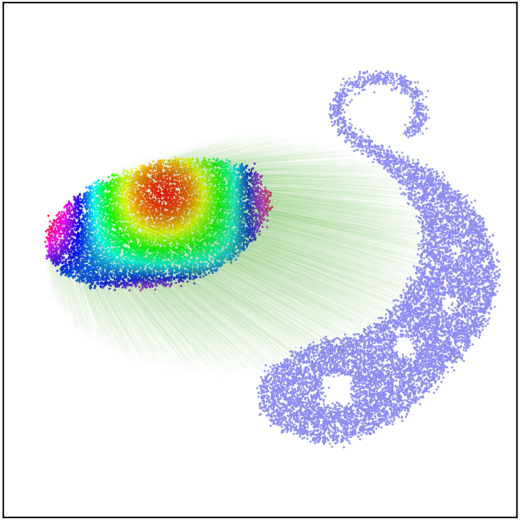
$$\text{Loss}(\alpha, \beta) \;=\; \tfrac{1}{2}\|g \star (\alpha - \beta)\|^2_{L^2(\mathbb{R}^D)} \;=\; \tfrac{1}{2}\langle\, \alpha - \beta\,,\, k \star (\alpha - \beta)\,\rangle$$

where $k \;=\; (g \circ (x \mapsto -x)) \,\star\, g\,.$

- Example: the Energy Distance, $k(x, y) = -\|x - y\|$:

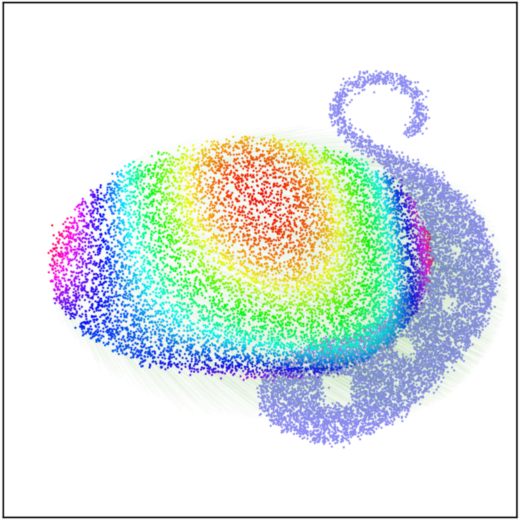
$$\begin{aligned}\text{Loss}(\alpha, \beta) \;&=\; \sum_i \sum_j \alpha_i \beta_j \,\|x_i - y_j\| \\ &\quad -\; \tfrac{1}{2} \sum_i \sum_j \alpha_i \alpha_j \,\|x_i - x_j\| \;-\; \tfrac{1}{2} \sum_i \sum_j \beta_i \beta_j \,\|y_i - y_j\|\,.\end{aligned}$$

Gradient flow as a toy registration problem



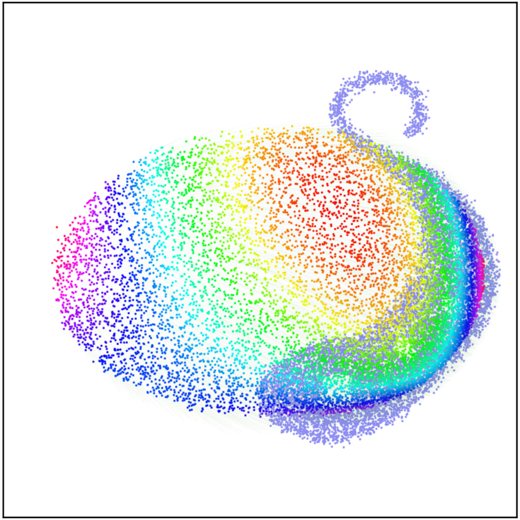
$t = .00$

Gradient flow as a toy registration problem



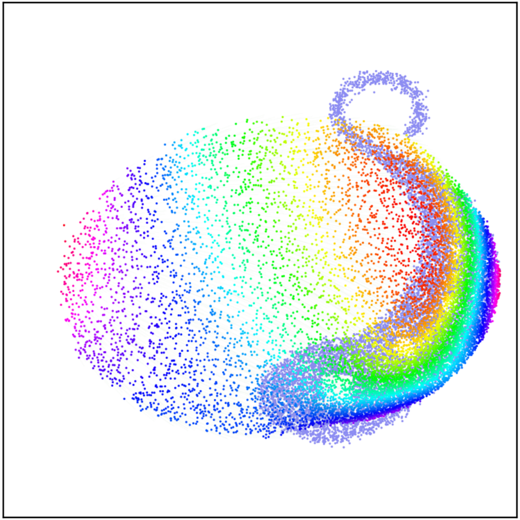
$t = .25$

Gradient flow as a toy registration problem



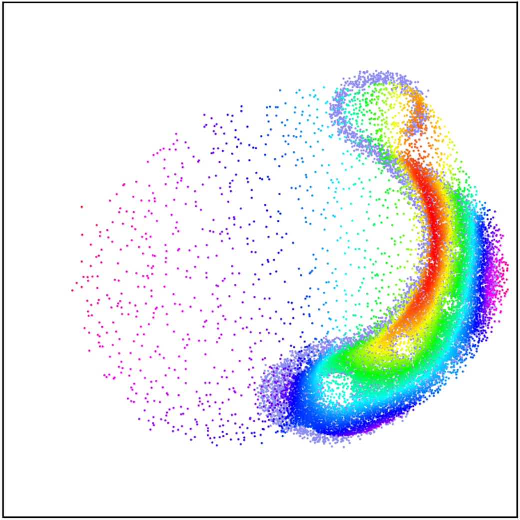
$t = .50$

Gradient flow as a toy registration problem



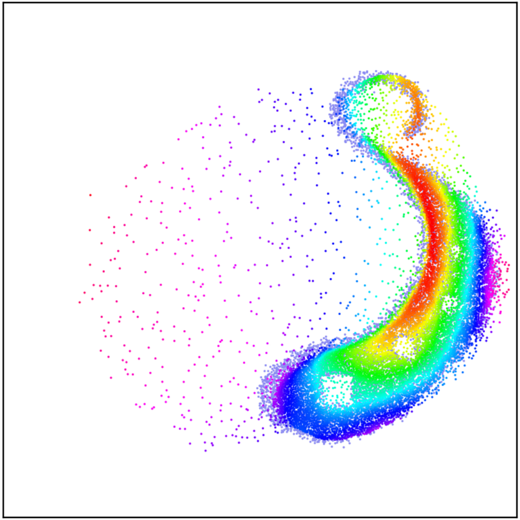
$t = 1.00$

Gradient flow as a toy registration problem



$t = 5.00$

Gradient flow as a toy registration problem

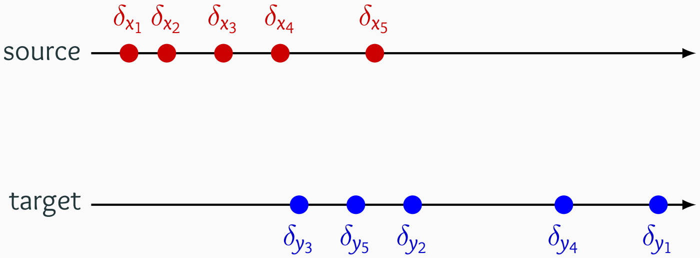


$t = 10.00$

The Wasserstein distance

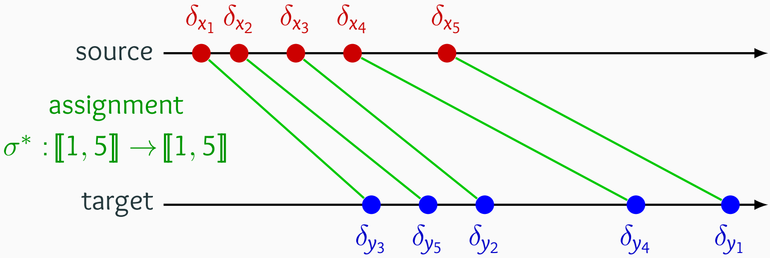
We need **clean gradients**, without artifacts. Let's **sort** our points.

Simple toy example in 1D :



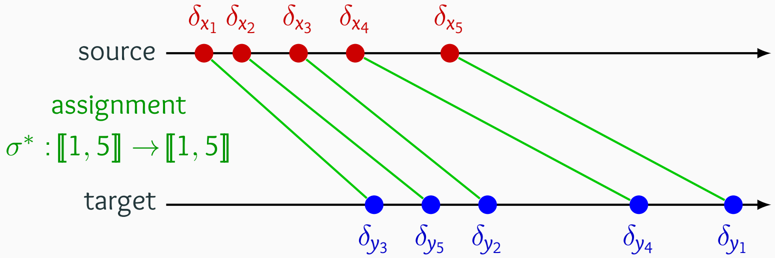
We need **clean gradients**, without artifacts. Let's **sort** our points.

Simple toy example in 1D:



We need **clean gradients**, without artifacts. Let's **sort** our points.

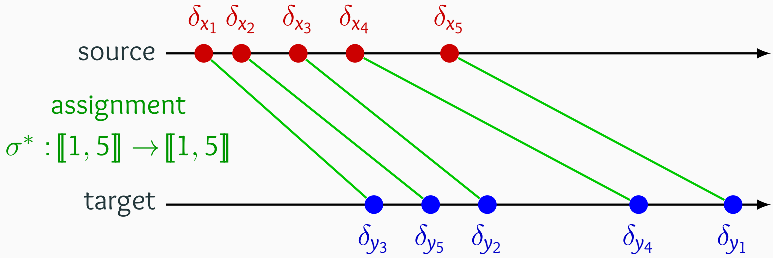
Simple toy example in 1D :



$$\text{OT}(\alpha, \beta) \;=\; \frac{1}{2N} \sum_{i=1}^{N} \left| x_i - y_{\sigma^*(i)} \right|^2$$

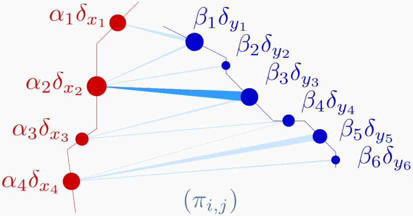
We need **clean gradients**, without artifacts. Let's **sort** our points.

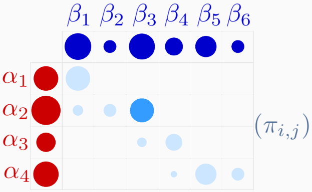
Simple toy example in 1D :



$$\mathrm{OT}(\alpha, \beta) \;=\; \frac{1}{2N} \sum_{i=1}^{N} \left| x_i - y_{\sigma^*(i)} \right|^2 \;=\; \min_{\sigma \in \mathcal{S}_N} \frac{1}{2N} \sum_{i=1}^{N} \left| x_i - y_{\sigma(i)} \right|^2$$

Optimal transport generalizes sorting to D > 1





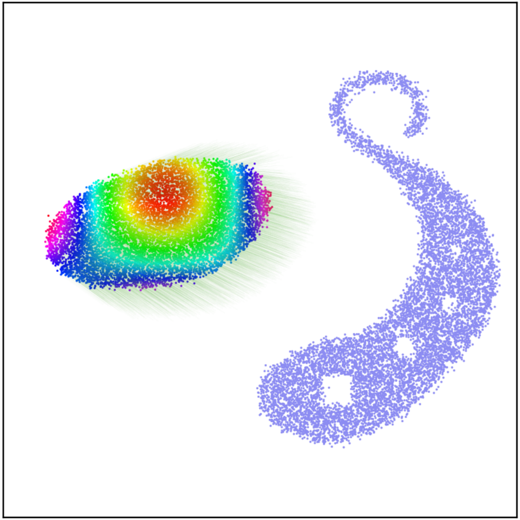
Minimize over N-by-M matrices (transport plans) π :

$$\mathrm{OT}(\alpha, \beta) \;=\; \min_{\pi} \underbrace{\sum_{i,j} \pi_{i,j} \cdot \tfrac{1}{2}|x_i - y_j|^2}_{\text{transport cost}}$$

subject to $\quad \pi_{i,j} \geqslant 0,$

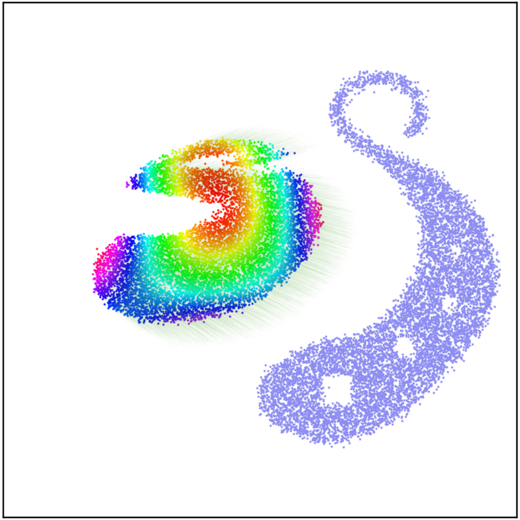
$$\sum_j \pi_{i,j} \;=\; \alpha_i, \quad \sum_i \pi_{i,j} \;=\; \beta_j.$$

The gradient of the Wasserstein distance is homogeneous



$t = .00$

The gradient of the Wasserstein distance is homogeneous

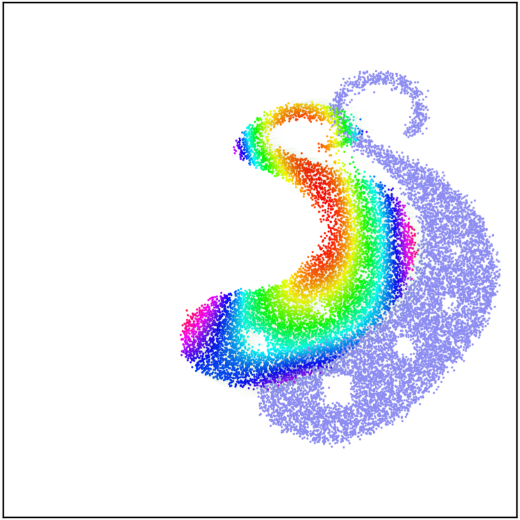


$t = .25$

The gradient of the Wasserstein distance is homogeneous

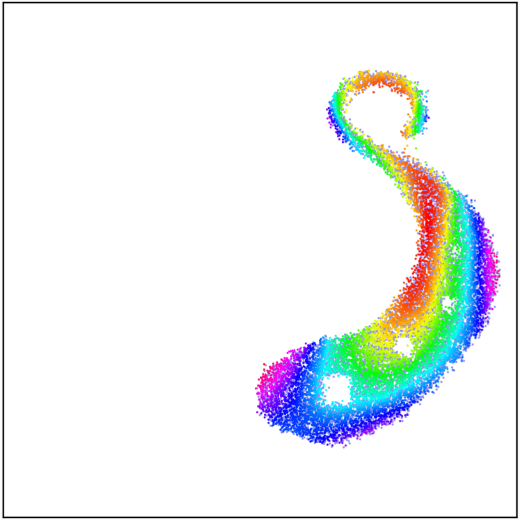
$t = .50$

The gradient of the Wasserstein distance is homogeneous



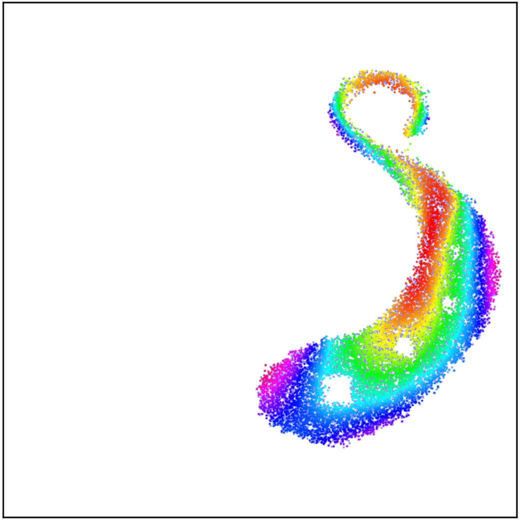
$t = 1.00$

The gradient of the Wasserstein distance is homogeneous



$t = 5.00$

The gradient of the Wasserstein distance is homogeneous



$t = 10.00$

Key properties [Bre91]

The Wasserstein loss $\mathrm{OT}(\alpha, \beta)$ is:

- **Symmetric:** $\mathrm{OT}(\alpha, \beta) = \mathrm{OT}(\beta, \alpha)\,.$
- **Positive:** $\mathrm{OT}(\alpha, \beta) \geqslant 0\,.$
- **Definite:** $\mathrm{OT}(\alpha, \beta) = 0 \iff \alpha = \beta\,.$
- **Translation-aware:** $\mathrm{OT}(\alpha,\ \mathrm{Translate}_{\vec{v}}(\alpha)\,) = \frac{1}{2}\|\,\vec{v}\,\|^2\,.$

- More generally, OT retrieves the unique **gradient of a convex function** $T = \nabla\varphi$ that maps α onto β :

$$\begin{aligned} &\text{In dimension 1,} && (x_i - x_j)\cdot(y_{\sigma(i)} - y_{\sigma(j)}) && \geqslant\ 0 \\ &\text{In dimension D,} && \langle\, x_i - x_j\ ,\ T(x_i) - T(x_j)\,\rangle_{\mathbb{R}^\mathrm{D}} && \geqslant\ 0\,. \end{aligned}$$

$\Longrightarrow$ Appealing generalization of an **increasing mapping.**

Can we scale this to real data?

Kantorovitch's dual formulation

$$\mathrm{OT}(\alpha,\beta) \;=\; \min_{\pi} \langle\, \pi \,,\, \mathrm{C}\,\rangle, \;\; \text{with } \mathrm{C}(x_i,y_j) \;=\; \tfrac{1}{p}\|x_i - y_j\|^p \quad\longrightarrow\; \text{Assignment}$$

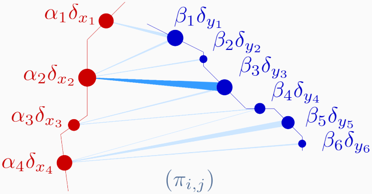
$$\text{s.t. } \pi \;\geqslant\; 0, \qquad \pi\,\mathbf{1} \;=\; \alpha, \qquad \pi^{\intercal}\mathbf{1} \;=\; \beta$$

Kantorovitch's dual formulation

$$\mathrm{OT}(\alpha, \beta) = \min_{\pi} \langle \pi , \mathrm{C} \rangle, \text{ with } \mathrm{C}(x_i, y_j) = \frac{1}{p}\|x_i - y_j\|^p \longrightarrow \text{Assignment}$$

$$\text{s.t. } \pi \geqslant 0, \quad \pi \mathbf{1} = \alpha, \quad \pi^\intercal \mathbf{1} = \beta$$

$\sum_{i,j} \pi_{i,j} \, \mathrm{C}(x_i, y_j)$

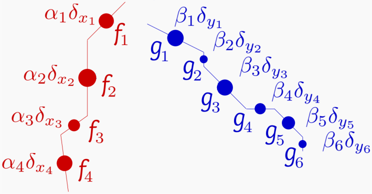


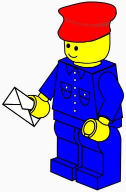
Kantorovitch's dual formulation

$$\mathrm{OT}(\alpha,\beta) \;=\; \min_{\pi} \langle\, \pi \,,\, \mathrm{C} \,\rangle, \;\; \text{with } \mathrm{C}(x_i,y_j) \;=\; \tfrac{1}{p}\|x_i - y_j\|^p \quad \longrightarrow \text{ Assignment}$$

$$\text{s.t. } \pi \;\geqslant\; 0, \quad \pi\,\mathbf{1} \;=\; \alpha, \quad \pi^{\intercal}\mathbf{1} \;=\; \beta$$

$\sum_{i,j} \pi_{i,j}\, \mathrm{C}(x_i,y_j)$

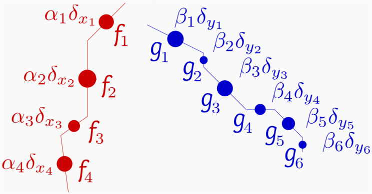


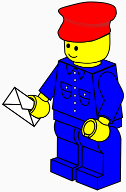


$\sum_i \alpha_i f_i + \sum_j \beta_j g_j$

$$\mathrm{OT}(\alpha,\beta) = \min_{\pi} \langle \pi , \mathrm{C} \rangle, \ \text{with } \mathrm{C}(x_i, y_j) = \frac{1}{p}\|x_i - y_j\|^p \quad \longrightarrow \text{Assignment}$$

$$\text{s.t. } \pi \geqslant 0, \quad \pi\,\mathbf{1} = \alpha, \quad \pi^\top \mathbf{1} = \beta$$





$\sum_{i,j} \pi_{i,j}\, \mathrm{C}(x_i, y_j)$

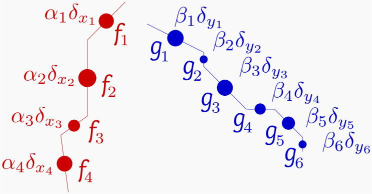
$\sum_i \alpha_i f_i + \sum_j \beta_j g_j$

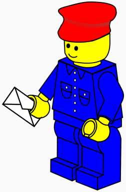
$$\max_{f,g} \quad \langle \alpha , f \rangle + \langle \beta , g \rangle \qquad \longrightarrow \text{FedEx}$$

$$\text{s.t.} \quad f(x_i) + g(y_j) \leqslant \mathrm{C}(x_i, y_j),$$

$$\mathrm{OT}(\alpha,\beta) = \min_{\pi} \langle \pi , \mathrm{C} \rangle, \text{ with } \mathrm{C}(x_i, y_j) = \frac{1}{p}\|x_i - y_j\|^p \quad \longrightarrow \text{Assignment}$$

$$\text{s.t. } \pi \geqslant 0, \quad \pi \mathbf{1} = \alpha, \quad \pi^{\top}\mathbf{1} = \beta$$





$\sum_{i,j} \pi_{i,j}\, \mathrm{C}(x_i, y_j)$

$\sum_i \alpha_i f_i + \sum_j \beta_j g_j$

$$= \max_{f,g} \quad \langle \alpha , f \rangle + \langle \beta , g \rangle \qquad \longrightarrow \text{FedEx}$$

$$\text{s.t.} \quad f(x_i) + g(y_j) \leqslant \mathrm{C}(x_i, y_j),$$

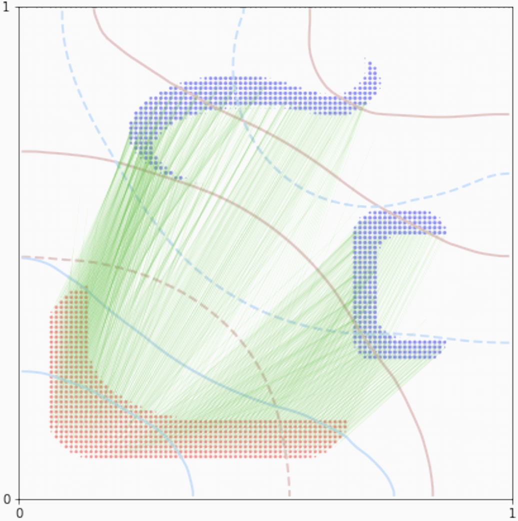
How should we solve the OT problem?

Key dates:

- [Kan42]: **Dual** problem, $O(N^2) \rightarrow O(N)$ memory footprint.
- [Kuh55]: **Hungarian** method in $O(N^3)$.
- [Ber79]: **Auction** algorithm in $O(N^2)$.
- [KY94]: **SoftAssign** = Sinkhorn + annealing, in $O(N^2)$.
- [GRL+98, CR00]: **Robust Point Matching** = Sinkhorn as a loss.
- [Cut13]: Start of the **GPU era**.
- [Mér11, Lév15, Sch19]: **Multiscale** CPU solvers in $O(N \log N)$.
- Today: **Multiscale Sinkhorn algorithm, on the GPU**.

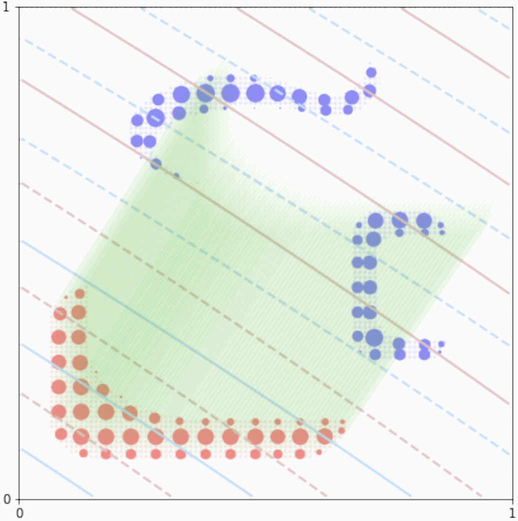
$\Longrightarrow$ Generalized **QuickSort** algorithm.

Visualizing F, G and the Brenier map $\nabla F(x_i) = -\frac{1}{\alpha_i}\partial_{x_i}\mathrm{OT}(\alpha, \beta)$



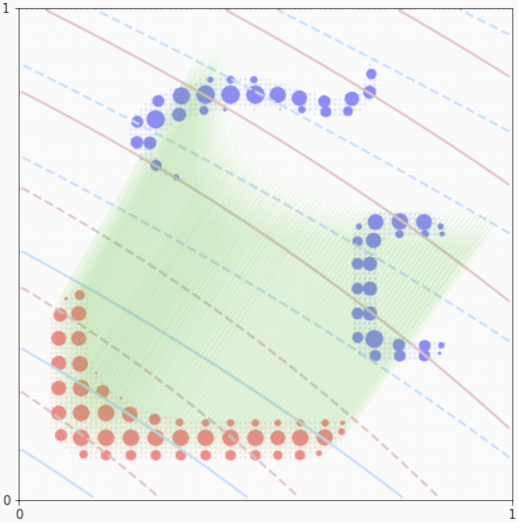
OT plan in 2D.

Visualizing F, G and the Brenier map $\nabla F(x_i) = -\frac{1}{\alpha_i}\partial_{x_i}\mathrm{OT}(\alpha, \beta)$



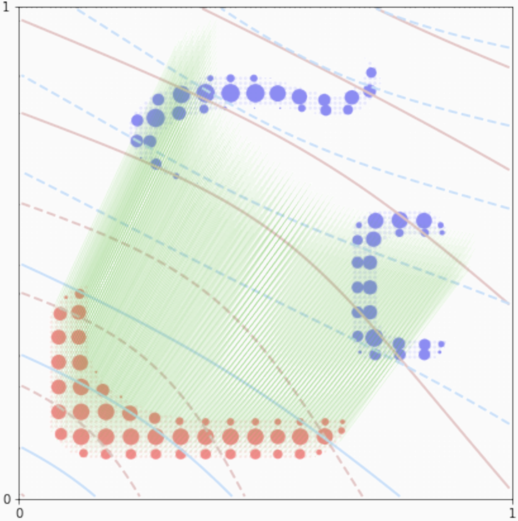
Iteration 0, blur = 2^0

Visualizing F, G and the Brenier map $\nabla F(x_i) = -\frac{1}{\alpha_i}\partial_{x_i}\mathrm{OT}(\alpha, \beta)$



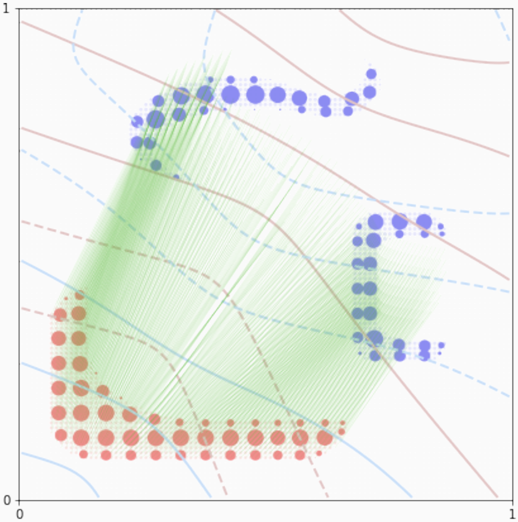
Iteration 1, blur = 2^{-1}

Visualizing F, G and the Brenier map $\nabla F(x_i) = -\frac{1}{\alpha_i}\partial_{x_i}\mathrm{OT}(\alpha, \beta)$



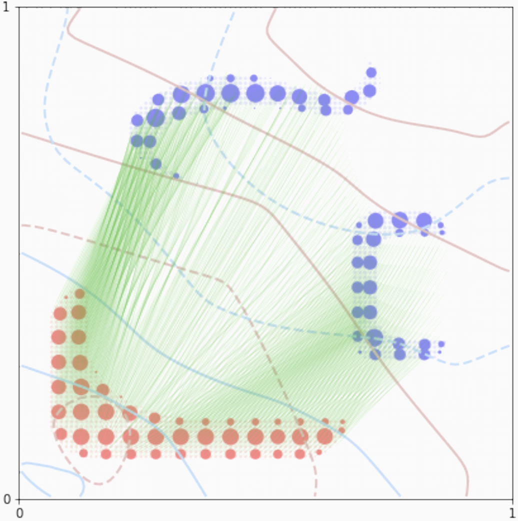
Iteration 2, blur = 2^{-2}

Visualizing F, G and the Brenier map $\nabla F(x_i) = -\frac{1}{\alpha_i}\partial_{x_i}\mathrm{OT}(\alpha, \beta)$



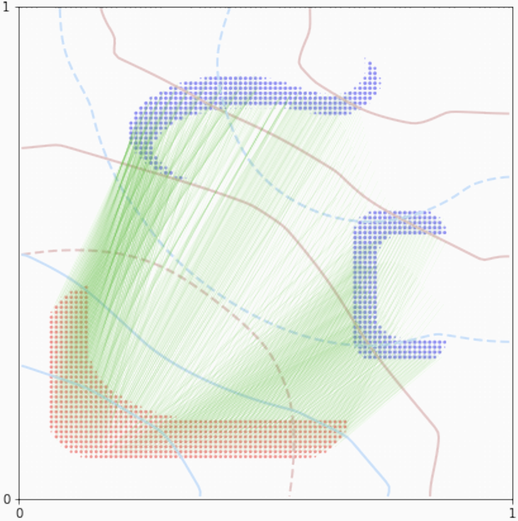
Iteration 3, blur = 2^{-3}

Visualizing F, G and the Brenier map $\nabla F(x_i) = -\frac{1}{\alpha_i}\partial_{x_i}\mathrm{OT}(\alpha, \beta)$



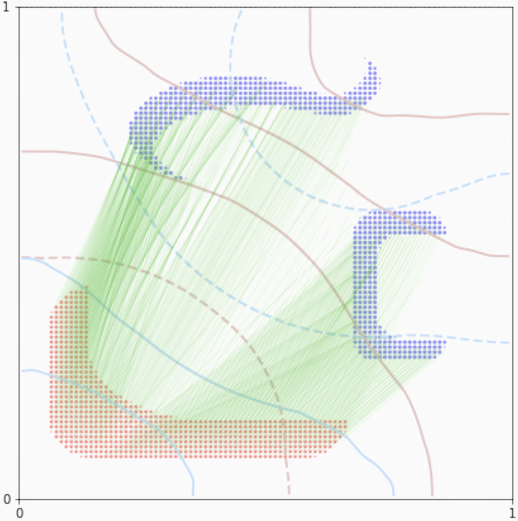
Iteration 4, blur = 2^{-4}

Visualizing F, G and the Brenier map $\nabla F(x_i) = -\frac{1}{\alpha_i}\partial_{x_i}\mathrm{OT}(\alpha, \beta)$



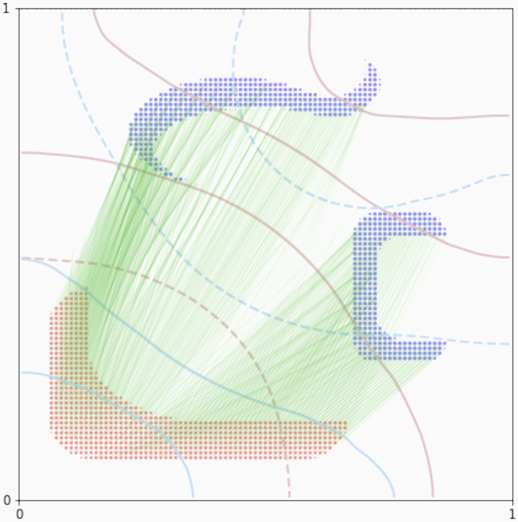
Iteration 5, blur = 2^{-5}

Visualizing F, G and the Brenier map $\nabla F(x_i) = -\frac{1}{\alpha_i}\partial_{x_i}\mathrm{OT}(\alpha, \beta)$



Iteration 6, blur = 2^{-6}

Visualizing F, G and the Brenier map $\nabla F(x_i) = -\frac{1}{\alpha_i}\partial_{x_i}\mathrm{OT}(\alpha, \beta)$



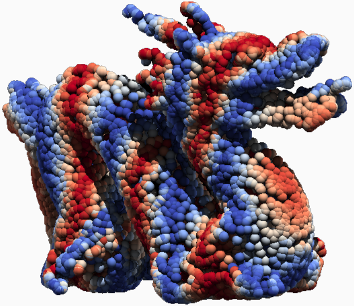
Iteration 7, blur = `.01`

Scaling up optimal transport to anatomical data

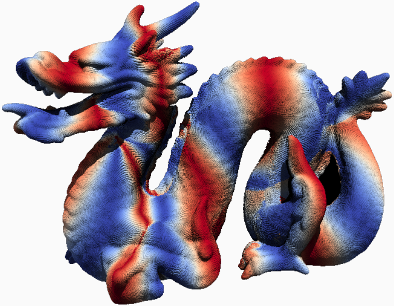
Progresses of the last decade add up to a $\times$ **100** - $\times$ **1000 acceleration:**

Sinkhorn GPU $\xrightarrow{\times 10}$ + KeOps $\xrightarrow{\times 10}$ + Annealing $\xrightarrow{\times 10}$ + Multiscale

With a precision of 1%, on a modern gaming GPU:



10k points in 30-50ms



100k points in 100-200ms

Geometric Loss functions for PyTorch

Our website: `www.kernel-operations.io/geomloss`

$\Longrightarrow$ `pip install geomloss` $\Longleftarrow$

```
# Large point clouds in [0,1]^3
import torch
x = torch.rand(100000, 3, requires_grad=True).cuda()
y = torch.rand(200000, 3).cuda()

# Define a Wasserstein loss between sampled measures
from geomloss import SamplesLoss
loss = SamplesLoss(loss="sinkhorn", p=2)
L = loss(x, y)  # By default, use constant weights
```

Soon: efficient support for **images**, **meshes** and generic metrics.

Optimal Transport, in practice

Wasserstein distance = Multi-dimensional sorting problem ?

The **three regimes** of Optimal Transport:

Wasserstein distance = Multi-dimensional sorting problem ?

The **three regimes** of Optimal Transport:

- α, β live in **dimension 1**:
  $\Longrightarrow$ Simple sorting problem.
  $\Longrightarrow$ Quicksort in $\mathbf{O}(\mathbf{N}\log\mathbf{N})$.

Wasserstein distance = Multi-dimensional sorting problem ?

The **three regimes** of Optimal Transport:

- α, β live in **dimension 1**:
  $\Longrightarrow$ Simple sorting problem.
  $\Longrightarrow$ Quicksort in $\mathbf{O}(\mathbf{N} \log \mathbf{N})$.

- α, β live in dimension **10+**:
  $\Longrightarrow$ The matrix of distances $\|x_i - y_j\|$ has very little structure.
  $\Longrightarrow$ Compute all pairs in $\geqslant \mathbf{O}(\mathbf{N}^2)$.

Wasserstein distance = Multi-dimensional sorting problem ?

The **three regimes** of Optimal Transport:

- α, β live in **dimension 1**:
  $\Longrightarrow$ Simple sorting problem.
  $\Longrightarrow$ Quicksort in $\mathbf{O(N \log N)}$.

- α, β have a **small** intrinsic **dimension**:
  $\Longrightarrow$ Rely on multiscale strategies.
  $\Longrightarrow$ Multiscale Sinkhorn in $\mathbf{O(N \log N)}$ **on the GPU.**

- α, β live in dimension **10+**:
  $\Longrightarrow$ The matrix of distances $\|x_i - y_j\|$ has very little structure.
  $\Longrightarrow$ Compute all pairs in $\geqslant \mathbf{O(N^2)}$.

Wasserstein distance = Multi-dimensional sorting problem ?

The **three regimes** of Optimal Transport:

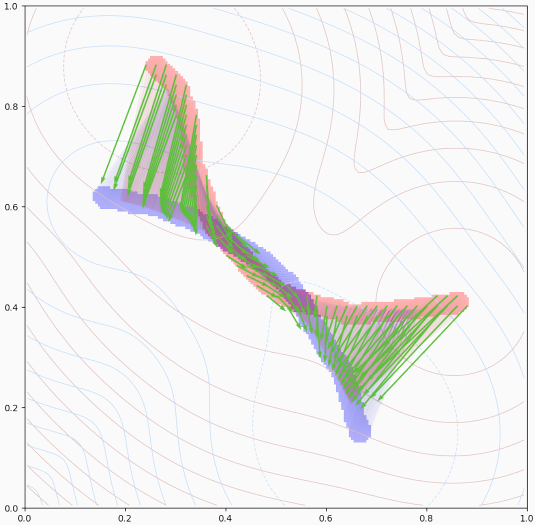
- α, β live in **dimension 1**:
  $\Longrightarrow$ Simple sorting problem.
  $\Longrightarrow$ Quicksort in $\mathbf{O(N \log N)}$.

- α, β have a **small** intrinsic **dimension**:
  $\Longrightarrow$ Rely on multiscale strategies.
  $\Longrightarrow$ Multiscale Sinkhorn in $\mathbf{O(N \log N)}$ **on the GPU.**

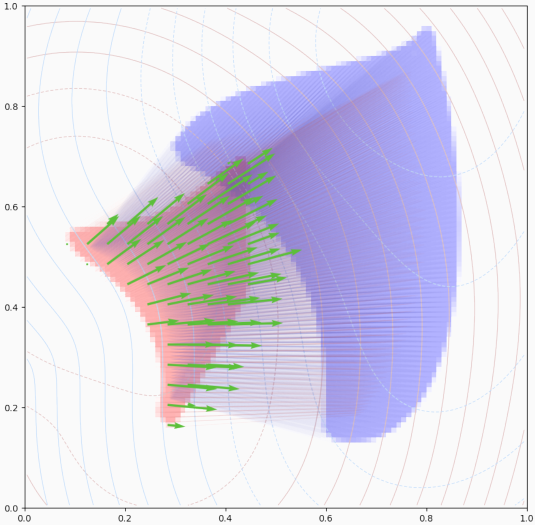
- α, β live in dimension **10+**:
  $\Longrightarrow$ The matrix of distances $\|x_i - y_j\|$ has very little structure.
  $\Longrightarrow$ Compute all pairs in $\geqslant \mathbf{O(N^2)}$.

$\Longrightarrow$ Multiscale Sinkhorn algorithm $\simeq$ Multi-dimensional **Quicksort.**

A global and geometric distance

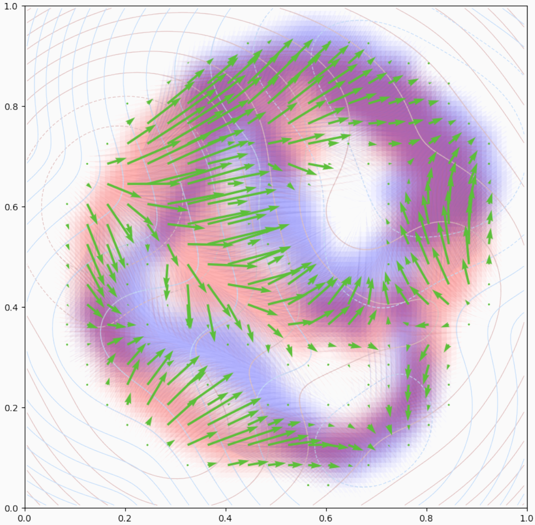


A high-quality gradient...



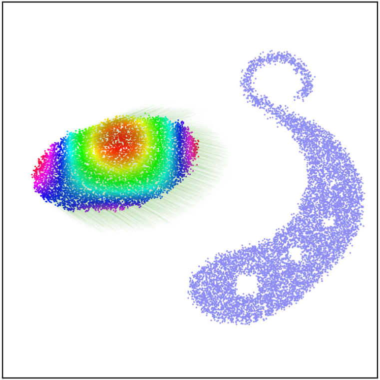
A high-quality gradient...

A global and geometric distance

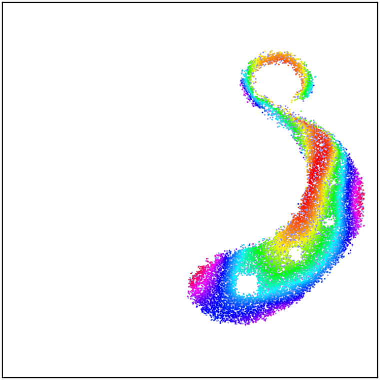


A high-quality gradient… But no preservation of topology!

Optimal transport = cheap'n easy registration? Beware!

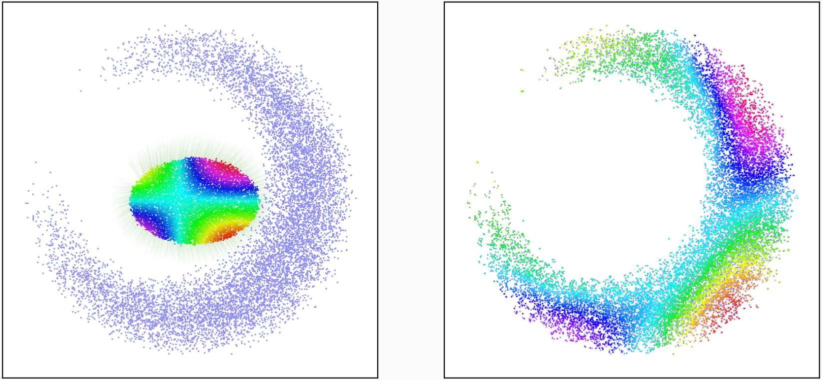


Before



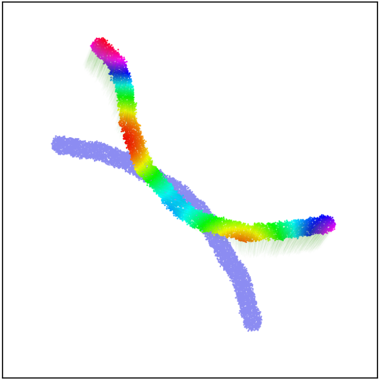
After

Optimal transport = cheap'n easy registration? Beware!



Before After

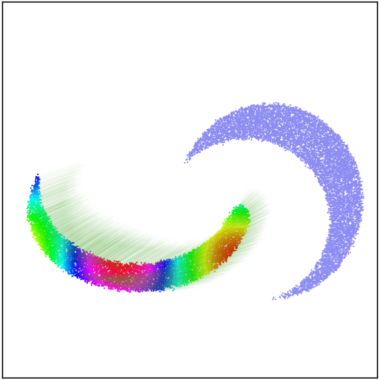
Optimal transport = cheap'n easy registration? Beware!



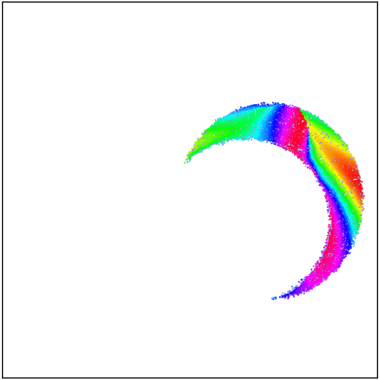
Before

After

Optimal transport = cheap'n easy registration? Beware!



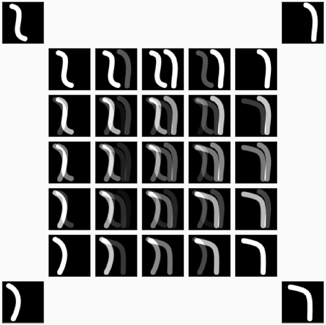
Before



After

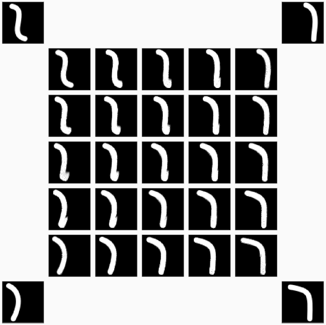
Affordable geometric interpolation [AC11]

$$\text{Barycenter } \alpha^* \;=\; \arg\min_{\alpha} \sum_{i=1}^{N} \lambda_i \,\text{Loss}(\,\alpha\,,\;\beta_i\,)\;.$$



Linear barycenters

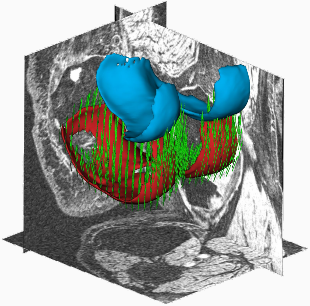
$\text{Loss}(\alpha, \beta) = \|\alpha - \beta\|^2_{L^2}$



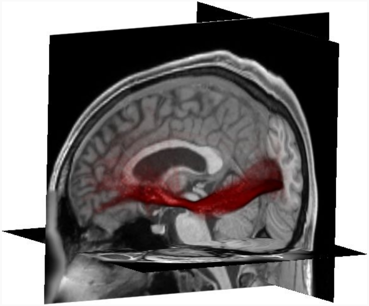
Wasserstein barycenters

$\text{Loss}(\alpha, \beta) = \text{OT}(\alpha, \beta)$

Applications to medical imaging

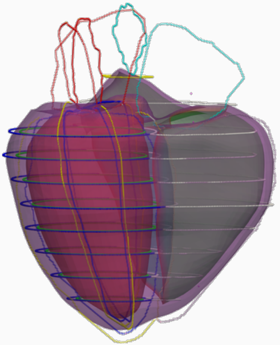


Knee caps

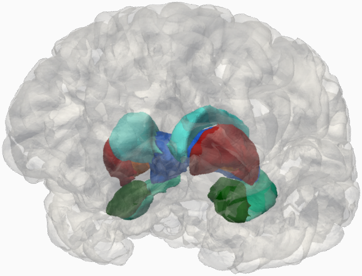


White matter bundles

Ongoing work: computational anatomy



Fast OT-based registration with
S. Joutard, X. Hao, A. Young from KCL,
Z. Shen, M. Niethammer from UNC.



Diffeomorphic and spline registration
e.g. Deformetrica LDDMM software
with the Aramis Inria team.

Going further: better features and/or topology-aware models

We now know how to soften the bijectivity constraints,
be **robust** to sampling noise and to some outliers.

But OT remains little more than **generalized sorting**, or **“nearest neighbour projection”** with a mass preservation **constraint**:

1. The **quality** of an OT matching is entirely driven by the matrix of cost values $C(x_i, y_j) = \|x_i - y_j\|^2$ – the cleaner, the better.

2. Guaranteeing the **preservation of topology** is very costly – as done for e.g. the diffeomorphic registration of brain images.

Going further: better features and/or topology-aware models

OT in 2D/3D with a squared Euclidean cost on the (x, y, z) coordinates is most relevant to e.g. fluid mechanics: it models the displacement of **non-viscous**, **incompressible** fluids.

This simple model may be good enough to track **small deformations** or ice motion between two neighboring frames in a "video".

For large displacements however, we use **domain-specific features and representations** that induce better cost matrices $C(x_i, y_j)$.

For instance, represent each iceberg as a single point with coordinates that correspond to geometric features such as the surface area, perimeter and geographic location?

As OT specialists, our main target is to enable the development of such "advanced" methods in the wider scientific community.

Scientific context, future works

Genuine team work


Alain Trouvé


Thibault Séjourné


F.-X. Vialard


Gabriel Peyré


Benjamin Charlier

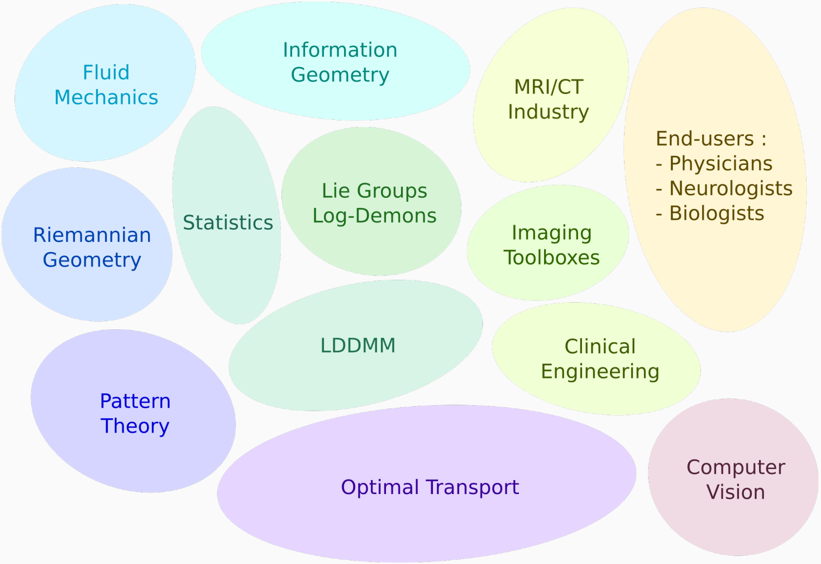

Joan Glaunès


Pierre Roussillon

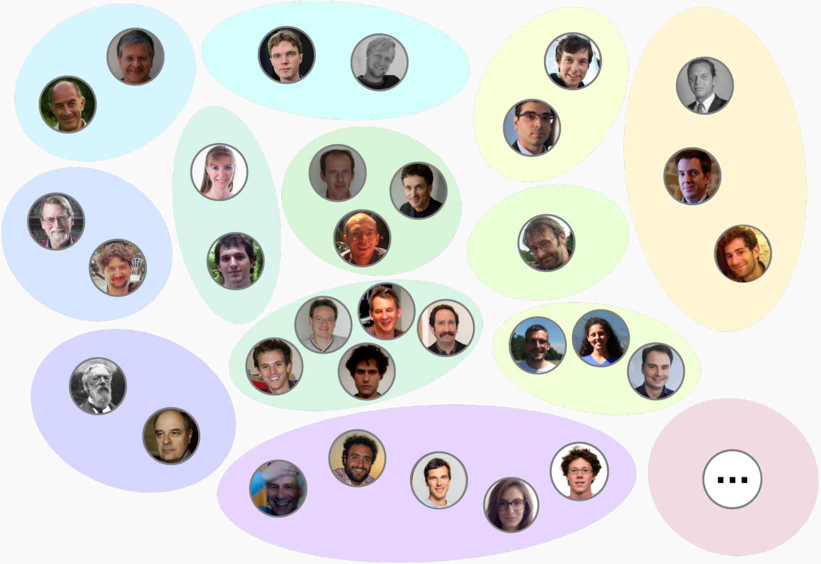

Pietro Gori

+ Freyr Sverrisson, Bruno Correia, Michael Bronstein, ...

Promoting cross-field interactions



Promoting cross-field interactions



The Python revolution

The emergence of an open and **modular** ecosystem of scientific tools has been a **boon** to the community.

Deep learning frameworks have put **GPU computing** and **automatic differentiation** in the hands of every student. (Incredible!)

These libraries have attracted significant backing from **industry** players (Google, Facebook, ...) and allowed the field to **boom** over the last decade.

The Python revolution

Interacting with other researchers, doctors
and engineers has never been so **easy**.

But on the other hand, PyTorch and TensorFlow have also **biased**
the field towards a **small set** of **well-supported** operations:
convolutions and matrix-matrix products, mostly.

This design choice is **not** due to an intrinsic limitation of GPUs:
our hardware is more than capable of **simulating** large,
open **3D worlds** in real-time!

As academic researchers, we must strive to keep **other paths open**.
Foster the development of a full range of methods,
from **robust** convex baselines
to **expressive** deep learning pipelines.

Our contribution to the community

KeOps and GeomLoss are:

- $+$ **Fast:** $\times 10$ - $\times 1{,}000$ speedup vs. naive GPU implementations.
- $+$ **Memory-efficient:** $O(\mathrm{N})$, not $O(\mathrm{N}^2)$.
- $+$ **Versatile, with a transparent interface**: freedom!
- $+$ **Powerful and well-documented:** research-friendly.
- $-$ Slow with **large feature vectors** of dimension $\mathrm{D} > 100$.

Our contribution to the community

KeOps and GeomLoss are:

- $+$ **Fast:** $\times 10$ - $\times 1{,}000$ speedup vs. naive GPU implementations.
- $+$ **Memory-efficient:** $O(\mathrm{N})$, not $O(\mathrm{N}^2)$.
- $+$ **Versatile, with a transparent interface:** freedom!
- $+$ **Powerful and well-documented:** research-friendly.
- $-$ Slow with **large feature vectors** of dimension $\mathrm{D} > 100$.

First half of 2021:

- $\rightarrow$ **Approximation strategies** (Nyström, etc.) in KeOps.
- $\rightarrow$ Wasserstein **barycenters** and **grid images** in GeomLoss.

An ongoing research project

Roadmap for KeOps + GeomLoss:

2017–18 **Proof of concept** with conference papers, online codes.
Get first feedback from the community.

2019–20 **Stable library** with solid theorems, a well-documented API.
KeOps backends for high-level packages.

2021–22 **Mature library** with focused application papers, full tutorials.
Works out-of-the-box for students and engineers.
$\Longrightarrow$ GeomLoss as a backend for POT v1.0.

2022+ **A standard toolbox**, with genuine clinical applications?
That's the target!

Conclusion

Key points

- **Symbolic** matrices are to **geometric** ML what
  sparse matrices are to **graph** processing:

  $\longrightarrow$ KeOps, **x30 speed-up** vs. PyTorch, TF and JAX.

  $\longrightarrow$ Useful in a wide range of settings.

- Optimal Transport = **generalized sorting**:

  $\longrightarrow$ Geometric gradients.

  $\longrightarrow$ Super-fast $O(N \log N)$ solvers.

- These tools open **new paths** for geometers and statisticians:

  $\longrightarrow$ GPUs are more **versatile** than you think.

  $\longrightarrow$ Ongoing work to provide **fast GPU backends** to researchers
  – going beyond what Google and Facebook are ready to pay for.

Conclusion

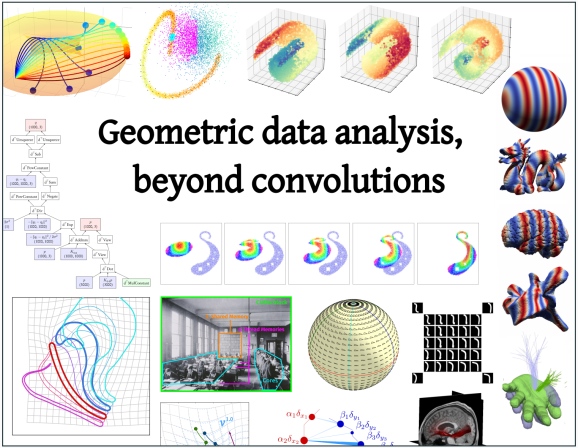
We believe that **KeOps** and **GeomLoss** will stimulate research on:

- **Clustering** methods: fast K-Means and EM iterations.
- Data **representation**: UMAP, fast KNN graphs with any metric.
- **Kernel** methods: kernel matrices.
- **Gaussian** processes: covariance matrices.
- **Geometric** deep learning: point convolutions.
- **Medical imaging**: computational anatomy.
- Geometric **statistics**: going beyond Euclidean models.
- Natural **language** processing: transformer networks?

What do you think?

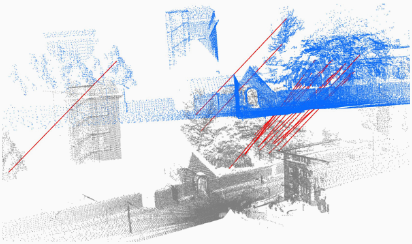
Documentation and tutorials are available online

$\Longrightarrow$ www.kernel-operations.io $\Longleftarrow$



www.jeanfeydy.com/geometric_data_analysis.pdf

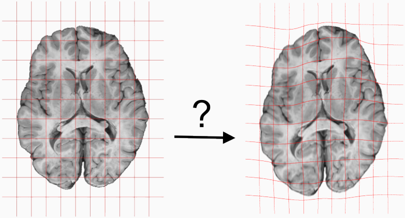
First setting: processing of point clouds



From the documentation of the Point Cloud Library.

- φ is **rigid** or affine
- Occlusions
- Outliers

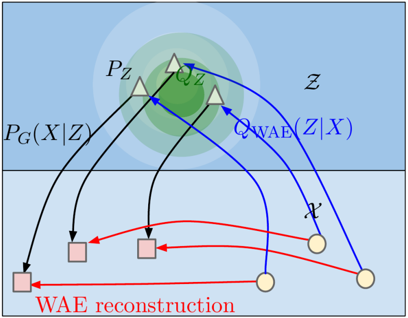
Second setting: medical imaging



From Marc Niethammer's Quicksilver slides.

- φ is a spline or a **diffeomorphism**
- Ill-posed problem
- Some occlusions

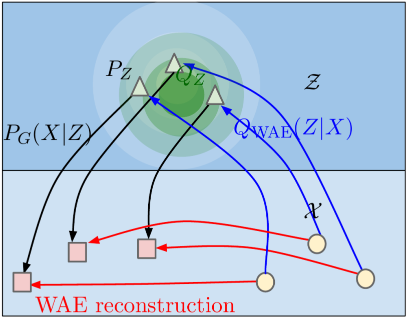
Third setting: training a generative model



Wasserstein Auto-Encoders,
Tolstikhin et al., 2018.

- φ is a **neural network**
- Very weak regularization
- High-dimensional space

Third setting: training a generative model



Wasserstein Auto-Encoders,
Tolstikhin et al., 2018.

- φ is a **neural network**
- Very weak regularization
- High-dimensional space

Which **Loss** function should we use?

Dual norms - link with the GANs literature

$$\text{Loss}(\alpha, \beta) \;=\; \max_{f \in B} \langle\, \alpha - \beta\,,\, f \rangle,$$

$$\text{look for}\;\; \theta^* \;=\; \arg\min_{\theta} \max_{f \in B} \langle\, \alpha(\theta) - \beta\,,\, f \rangle$$

Dual norms - link with the GANs literature

$$\mathrm{Loss}(\alpha, \beta) \;=\; \max_{f \in B} \langle\, \alpha - \beta\,,\, f \rangle,$$

$$\text{look for } \theta^* \;=\; \arg\min_{\theta} \max_{f \in B} \langle\, \alpha(\theta) - \beta\,,\, f \rangle$$

- $B = \{\, \| f \|_\infty \leqslant 1 \,\} \implies$ Loss $=$ TV norm:
  - zero geometry
  - **too many** test functions

Dual norms - link with the GANs literature

$$\text{Loss}(\alpha, \beta) \;=\; \max_{f \in B} \langle\, \alpha - \beta\,,\, f\,\rangle,$$

$$\text{look for } \;\theta^* \;=\; \arg\min_{\theta} \max_{f \in B} \langle\, \alpha(\theta) - \beta\,,\, f\,\rangle$$

- $B = \{\, \|\, f\,\|_\infty \leqslant 1\,\} \implies$ Loss $=$ TV norm:
    - zero geometry
    - **too many** test functions

- $B = \{\, \|\, f\,\|_2^2 + \|\, \nabla f\,\|_2^2 + \cdots \leqslant 1\,\} \implies$ Loss $=$ kernel norm:
    - may saturate at infinity
    - **screening** artifacts

Dual norms - link with the GANs literature

$$\text{Loss}(\alpha, \beta) \;=\; \max_{f \in B} \langle\, \alpha - \beta \,,\, f \rangle,$$

$$\text{look for } \theta^* \;=\; \arg\min_{\theta} \max_{f \in B} \langle\, \alpha(\theta) - \beta \,,\, f \rangle$$

- $B = \{\, f \text{ is 1-Lipschitz} \,\} \implies$ Loss $=$ Wasserstein-1 (OT_0):

Dual norms - link with the GANs literature

$$\text{Loss}(\alpha, \beta) = \max_{f \in B} \langle \alpha - \beta, f \rangle,$$

$$\text{look for } \theta^* = \arg\min_{\theta} \max_{f \in B} \langle \alpha(\theta) - \beta, f \rangle$$

- $B = \{ f \text{ is 1-Lipschitz} \} \implies$ Loss $=$ Wasserstein-1 (OT_0):
    - S_ε is nearly as efficient as a **closed formula**

Dual norms - link with the GANs literature

$$\text{Loss}(\alpha, \beta) \;=\; \max_{f \in B} \langle\, \alpha - \beta \,,\, f \,\rangle,$$

$$\text{look for } \;\theta^* \;=\; \arg\min_{\theta} \max_{f \in B} \langle\, \alpha(\theta) - \beta \,,\, f \,\rangle$$

- $B = \{\, f \text{ is 1-Lipschitz} \,\} \implies$ Loss $=$ Wasserstein-1 (OT_0):
  - S_ε is nearly as efficient as a **closed formula**
  - relevant in **low dimensions**
  - **useless** in $(\mathbb{R}^{512\times 512}, \|\cdot\|_2)$: the ground cost makes no sense

Dual norms - link with the GANs literature

$$\mathrm{Loss}(\alpha, \beta) \;=\; \max_{f \in B} \langle\, \alpha - \beta \,,\, f \,\rangle,$$

$$\text{look for } \theta^* \;=\; \arg\min_{\theta} \max_{f \in B} \langle\, \alpha(\theta) - \beta \,,\, f \,\rangle$$

- $B = \{\, f \text{ is 1-Lipschitz} \,\} \implies \mathrm{Loss} = \text{Wasserstein-1 } (\mathrm{OT}_0)$:
  - S_ε is nearly as efficient as a **closed formula**
  - relevant in **low dimensions**
  - **useless** in $(\mathbb{R}^{512\times 512}, \|\cdot\|_2)$: the ground cost makes no sense

- $B \simeq \{\, f \text{ is 1-Lipschitz} \,\} \bigcap \{\, f \text{ is a } \textbf{CNN} \,\}$

  $\implies \mathrm{Loss} = \text{Wasserstein GAN}:$

Dual norms - link with the GANs literature

$$\text{Loss}(\alpha, \beta) \;=\; \max_{f \in B} \langle\, \alpha - \beta \,,\, f \,\rangle,$$

$$\text{look for } \;\theta^* \;=\; \arg\min_{\theta} \max_{f \in B} \langle\, \alpha(\theta) - \beta \,,\, f \,\rangle$$

- $B = \{\, f \text{ is 1-Lipschitz} \,\} \implies$ Loss $=$ Wasserstein-1 (OT_0):
  - S_ε is nearly as efficient as a **closed formula**
  - relevant in **low dimensions**
  - **useless** in $(\mathbb{R}^{512\times 512}, \|\cdot\|_2)$: the ground cost makes no sense

- $B \simeq \{\, f \text{ is 1-Lipschitz} \,\} \bigcap \{\, f \text{ is a } \textbf{CNN} \,\}$
  $\implies$ Loss $=$ Wasserstein GAN :
  - use **perceptually sensible** test functions

Dual norms - link with the GANs literature

$$\text{Loss}(\alpha, \beta) = \max_{f \in B} \langle \alpha - \beta , f \rangle,$$

$$\text{look for } \theta^* = \arg\min_{\theta} \max_{f \in B} \langle \alpha(\theta) - \beta , f \rangle$$

- $B = \{ f \text{ is 1-Lipschitz} \} \implies \text{Loss} = \text{Wasserstein-1 } (\text{OT}_0)$:
  - S_ε is nearly as efficient as a **closed formula**
  - relevant in **low dimensions**
  - **useless** in $(\mathbb{R}^{512\times 512}, \|\cdot\|_2)$: the ground cost makes no sense

- $B \simeq \{ f \text{ is 1-Lipschitz} \} \bigcap \{ f \text{ is a } \textbf{CNN} \}$
  $\implies \text{Loss} = \text{Wasserstein GAN}$:
  - use **perceptually sensible** test functions
  - no simple formula: use **gradient ascent**

Dual norms - link with the GANs literature

$$\text{Loss}(\alpha, \beta) \;=\; \max_{f \in B} \langle\, \alpha - \beta \,,\, f \,\rangle,$$

$$\text{look for } \; \theta^* \;=\; \arg\min_{\theta} \max_{f \in B} \langle\, \alpha(\theta) - \beta \,,\, f \,\rangle$$

- $B = \{\, f \text{ is 1-Lipschitz} \,\} \implies$ Loss $=$ Wasserstein-1 (OT_0):
  - S_ε is nearly as efficient as a **closed formula**
  - relevant in **low dimensions**
  - **useless** in $(\mathbb{R}^{512\times 512}, \|\cdot\|_2)$: the ground cost makes no sense

- $B \simeq \{\, f \text{ is 1-Lipschitz} \,\} \bigcap \{\, f \text{ is a } \textbf{CNN} \,\}$
  $\implies$ Loss $=$ Wasserstein GAN :
  - use **perceptually sensible** test functions
  - no simple formula: use **gradient ascent**
  - can we provide relevant **insights** to the ML community?

References i

M. Agueh and G. Carlier.
Barycenters in the Wasserstein space.
SIAM Journal on Mathematical Analysis, 43(2):904–924, 2011.

John Ashburner.
A fast diffeomorphic image registration algorithm.
Neuroimage, 38(1):95–113, 2007.

Dimitri P Bertsekas.
A distributed algorithm for the assignment problem.
Lab. for Information and Decision Systems Working Paper, M.I.T., Cambridge, MA, 1979.

Y. Brenier.
Polar factorization and monotone rearrangement of vector-valued functions.
Comm. Pure Appl. Math., 44(4):375–417, 1991.

Christophe Chnafa, Simon Mendez, and Franck Nicoud.
Image-based large-eddy simulation in a realistic left heart.
Computers & Fluids, 94:173–187, 2014.

Haili Chui and Anand Rangarajan.
A new algorithm for non-rigid point matching.
In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 2, pages 44–51. IEEE, 2000.

Adam Conner-Simons and Rachel Gordon.
Using ai to predict breast cancer and personalize care.
`http://news.mit.edu/2019/using-ai-predict-breast-cancer-and-personalize-c`
2019.
MIT CSAIL.

Marco Cuturi.
Sinkhorn distances: Lightspeed computation of optimal transport.
In *Advances in Neural Information Processing Systems*, pages 2292–2300, 2013.

References iv

Olivier Ecabert, Jochen Peters, Matthew J Walker, Thomas Ivanc, Cristian Lorenz, Jens von Berg, Jonathan Lessick, Mani Vembar, and Jürgen Weese.
Segmentation of the heart and great vessels in CT images using a model-based adaptation framework.
Medical image analysis, 15(6):863–876, 2011.

Joan Glaunes.
Transport par difféomorphismes de points, de mesures et de courants pour la comparaison de formes et l'anatomie numérique.
These de sciences, Université Paris, 13, 2005.

Steven Gold, Anand Rangarajan, Chien-Ping Lu, Suguna Pappu, and Eric Mjolsness.
New algorithms for 2d and 3d point matching: Pose estimation and correspondence.
Pattern recognition, 31(8):1019–1031, 1998.

Leonid V Kantorovich.
On the translocation of masses.
In *Dokl. Akad. Nauk. USSR (NS)*, volume 37, pages 199–201, 1942.

Irene Kaltenmark, Benjamin Charlier, and Nicolas Charon.
A general framework for curve and surface comparison and registration with oriented varifolds.
In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

Harold W Kuhn.
The Hungarian method for the assignment problem.
Naval research logistics quarterly, 2(1-2):83–97, 1955.

Jeffrey J Kosowsky and Alan L Yuille.
The invisible hand algorithm: Solving the assignment problem with statistical physics.
Neural networks, 7(3):477–490, 1994.

Bruno Lévy.
A numerical algorithm for l2 semi-discrete optimal transport in 3d.
ESAIM: Mathematical Modelling and Numerical Analysis, 49(6):1693–1715, 2015.

Christian Ledig, Andreas Schuh, Ricardo Guerrero, Rolf A Heckemann, and Daniel Rueckert.
Structural brain imaging in Alzheimer's disease and mild cognitive impairment: biomarker analysis and shared morphometry database.
Scientific reports, 8(1):11258, 2018.

Quentin Mérigot.
A multiscale approach to optimal transport.
In *Computer Graphics Forum*, volume 30, pages 1583–1592. Wiley Online Library, 2011.

Ptrump16.
Irm picture.
`https://commons.wikimedia.org/w/index.php?curid=64157788`, 2019.
CC BY-SA 4.0.

Bernhard Schmitzer.
Stabilized sparse scaling algorithms for entropy regularized transport problems.
SIAM Journal on Scientific Computing, 41(3):A1443–A1481, 2019.

Thibault Séjourné, Jean Feydy, François-Xavier Vialard, Alain Trouvé, and Gabriel Peyré.
Sinkhorn divergences for unbalanced optimal transport.
arXiv preprint arXiv:1910.12958, 2019.