

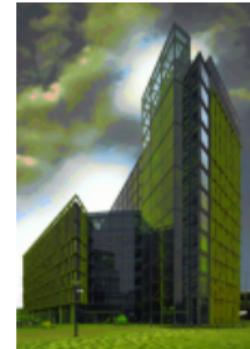
# A hands-on tutorial on Optimal Transport



Felipe Tobar

Department of Mathematics & I-X  
Imperial College London

22 May, 2025



[github.com/felipe-tobar/OT-tutorial](https://github.com/felipe-tobar/OT-tutorial)

# Overview

① Introduction

② Part I

The Optimal Transport Problem

③ Part II

The Wasserstein distance and metric properties

④ Part III (if time allows)

OT for time series: The Wasserstein-Fourier distance

## Preliminary remarks

- Based on the Tutorial **Optimal Transport for Signal Processing** given at IEEE MLSP 2024 w/ Laetitia Chapel (IRISA)
- Examples based on **POT: Python Optimal Transport Toolbox** ([pythonot.github.io](https://pythonot.github.io))
- Tutorial repository: [github.com/felipe-tobar/OT-tutorial](https://github.com/felipe-tobar/OT-tutorial)
- To clone the tutorial's conda environment, run: (tested on OSX and Linux):

```
conda env create -f environment.yml
```

∞ **thanks to:** Elsa Cazelles (Toulouse), Fernando Fleitas (U. Chile), Marco Cuturi (Apple/ENSAE), Rémi Flamary (École Polytechnique)

# OT for data analysis in a nutshell

Optimal Transportation theory is a set of tools for computing distances between distributions



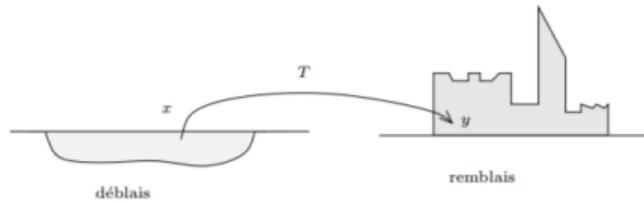
We are interested in OT because

- it gives a geometric-based view of probability distributions
- it enhances the machine learning (generative models), and data analysis (histograms) toolboxes.
- it addresses shortcomings of other divergences
- it is beautiful (but that's subjective)

Figure taken from <https://www.microsoft.com/en-us/research/blog/measuring-dataset-similarity-using-optimal-transport/>.  
Credit: David Alvarez-Melis & Nicolo Fusi (Microsoft Research)

# Origins of OT: Old and New

**Gaspard Monge (1781):** How to transport a pile of sand onto a hole in an optimal way?



**Leonid Kantorovich (1942):** Generalisation of Monge's and dual formulation

**Yann Brenier (1987):** Connections w/ PDEs, fluid mechanics, probability theory, and more.

**A number of Fields Medals (2010+)** Villani, Hairer, Figalli

**Marco Cuturi (2013):** Breakthrough in Machine Learning

# Part I

## The Optimal Transport Problem

**CUARTEL C2**

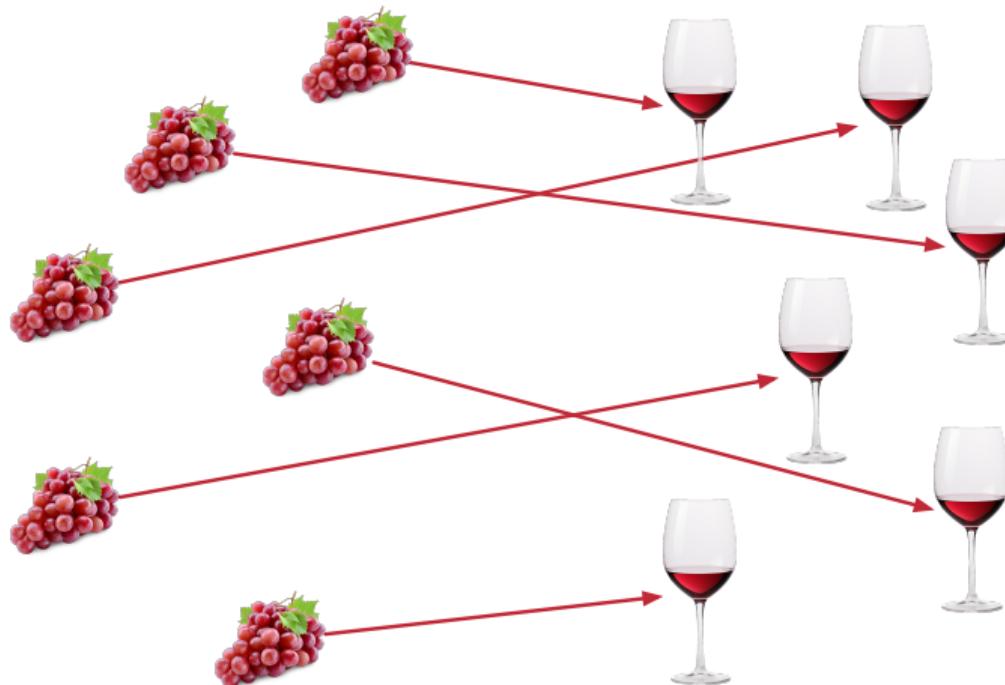
**CARMÉNÉRE**

**2.1 HA . 1994**

**2.3\* 1.2 - 90 H**



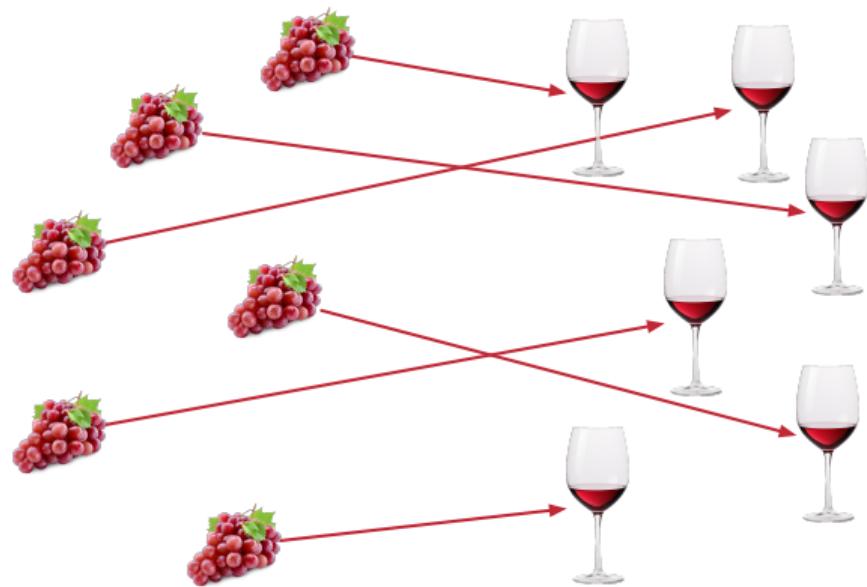
# The assignment problem



# The assignment problem: encoding the real world

- Weighted masses
- Different number of sources/targets
- Straight path is not possible
- New source/target becomes available

(we'll start with a simple case)



# Monge formulation<sup>1</sup>

**Objective:** To move a pile of mass from one location to another at a minimum effort

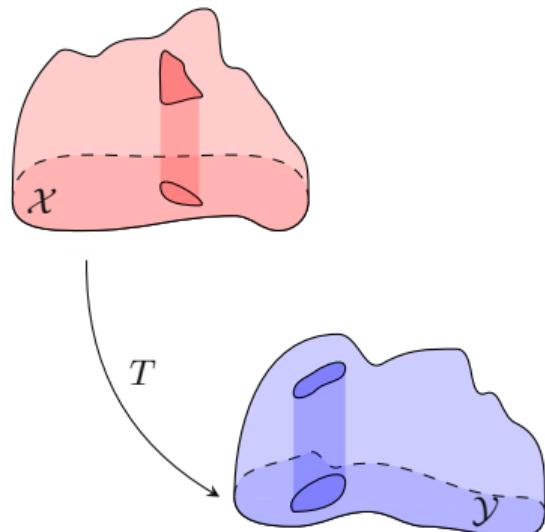
**Notation:**

- **Piles of mass** are probability distributions,  $\mu$  and  $\nu$ , corresponding to random variables  $X \in \mathcal{X}$  and  $Y \in \mathcal{Y}$ .
- **Moving procedure** is a function  $T : x \in \mathcal{X} \mapsto y \in \mathcal{Y}$ .
- **Moving cost** encoded as  $c : (x, y) \in \mathcal{X} \times \mathcal{Y} \mapsto c(x, y) \in \mathbb{R}$ .

**Optimise** the total transport cost

$$\text{OT}(\mu, \nu) = \min_{T \in M_{X,Y}} \sum_{x \in \mathcal{X}} c(x, T(x)), \quad (1)$$

where  $M_{X,Y} = \{T : \mathcal{X} \rightarrow \mathcal{Y}, \text{ s.t., } T_{\#}\mu = \nu\}$  is the space of *admissible transport maps*.



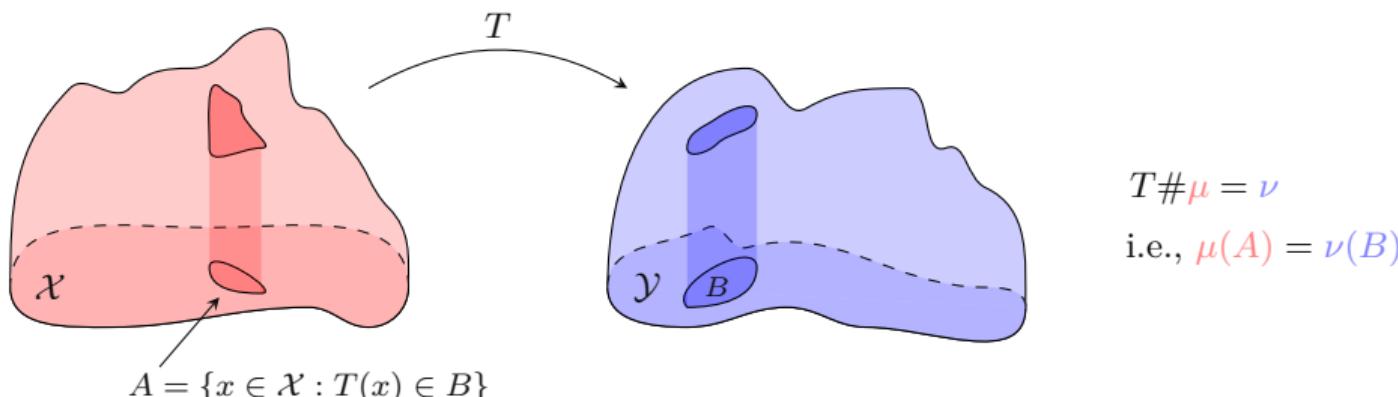
<sup>1</sup>Monge, G. (1781). Mémoire sur la théorie des déblais et des remblais. De l'Imprimerie Royale.

# The transport map (aka the *pushforward* operator $T_{\#}$ )

$T$  transports mass from  $\mathcal{X}$  to  $\mathcal{Y}$ , meaning that for any subset  $A \in \mathcal{Y}$ , one has

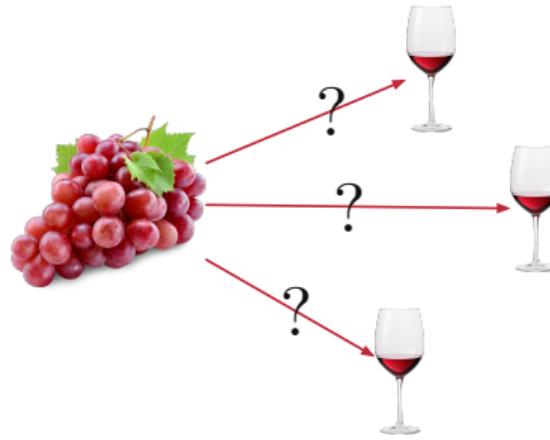
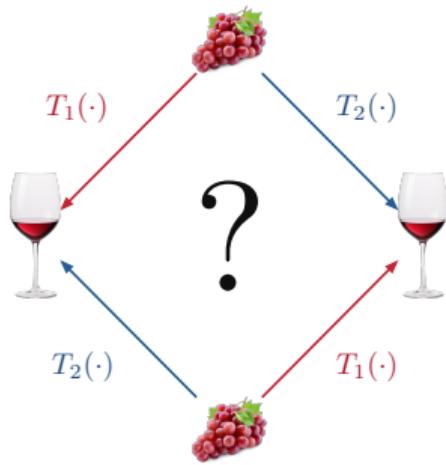
$$\nu(A) = \mu(T^{-1}(A)), \quad (2)$$

where  $T^{-1}(A) = \{x \in \mathcal{X}, s.t. T(x) \in A\}$  is the preimage of  $A$  under  $T$ .



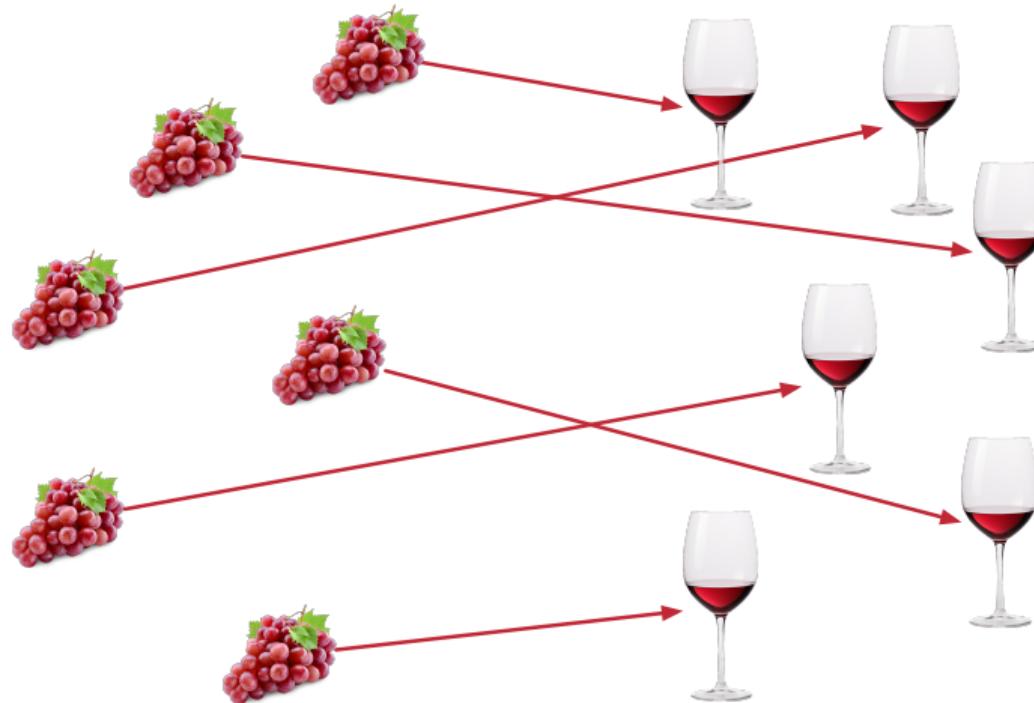
(figure adapted from: Matthew Thorpe, *Introduction to Optimal Transport* 2018)

## Warning: Neither existence nor uniqueness is guaranteed

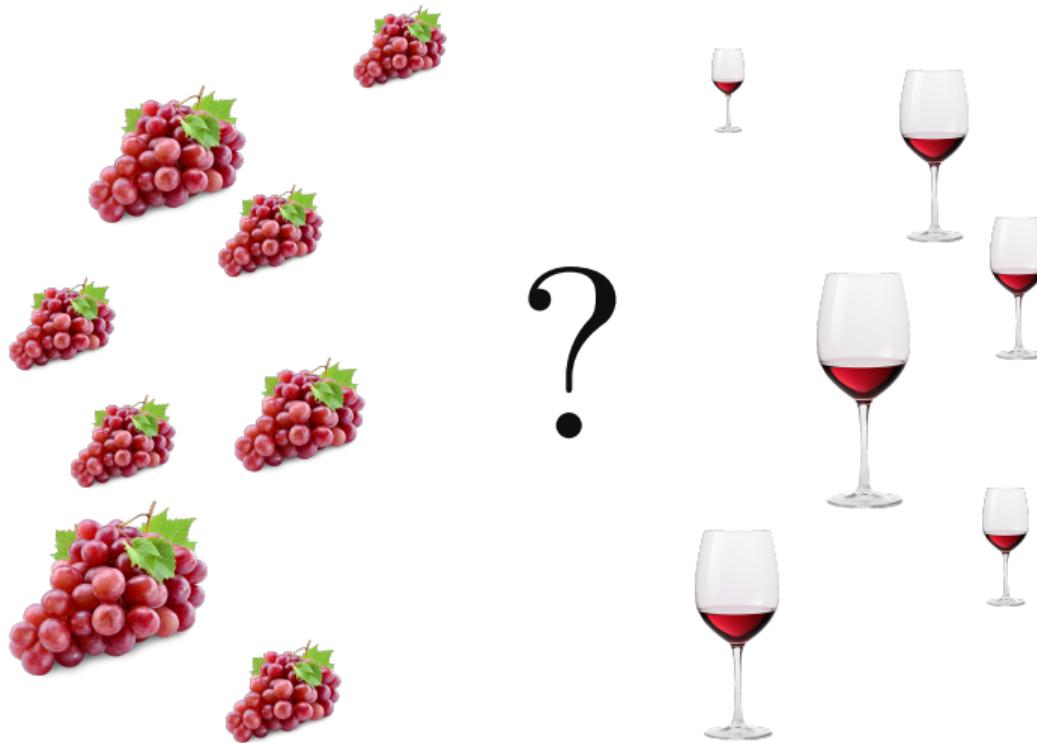


**Remark:** When both distributions have the same number of same-weight atoms, e.g., pixels or class instances, Monge's formulation can be solved. However, when dealing with different number of *weighted samples*, **Monge's map might be unable to transport the mass.**

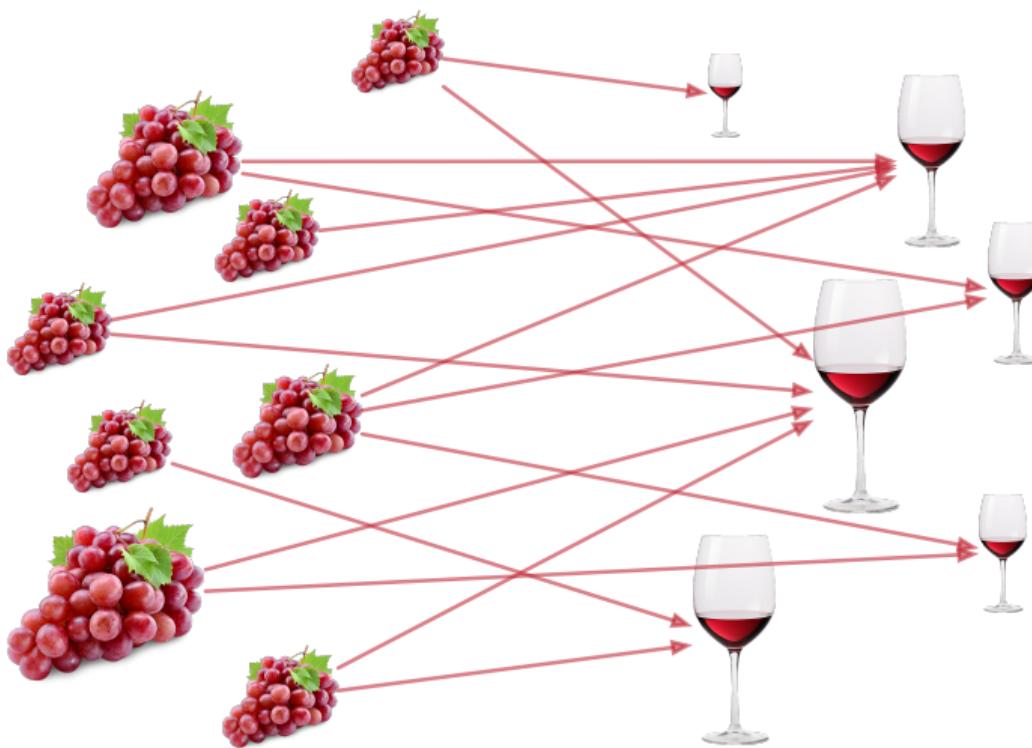
# Kantorovich relaxation: mass splitting



# Kantorovich relaxation: mass splitting



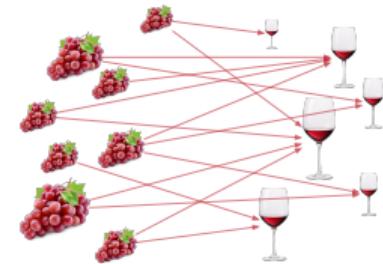
# Kantorovich relaxation: mass splitting



# Kantorovich's *transport plan*

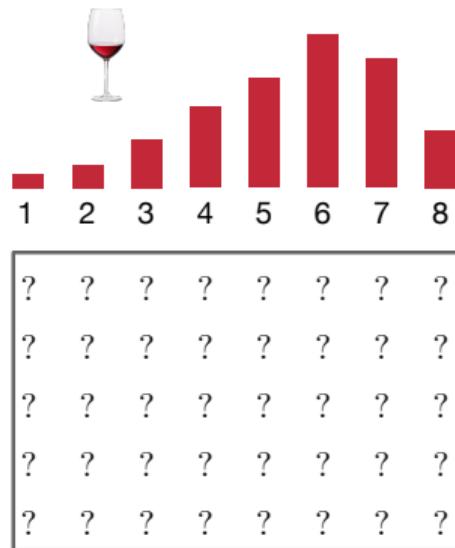
$$\text{OT}(\mu, \nu) = \inf_{P \in \Pi_{\mu, \nu}} \langle P, C \rangle = \sum_{i,j}^{n,m} C_{ij} P_{ij}$$

where  $\Pi_{\mu, \nu} = \{P \in [0, 1]^{m \times n} : \sum_{i=1}^m P_{ij} = \nu_j, \sum_{j=1}^n P_{ij} = \mu_i\}$



Transport plan

$P$



Cost Matrix

$C$

	1	2	3	4	5	6	7	8
1	\$	\$	\$	\$	\$	\$	\$	\$
2	\$	\$	\$	\$	\$	\$	\$	\$
3	\$	\$	\$	\$	\$	\$	\$	\$
4	\$	\$	\$	\$	\$	\$	\$	\$
5	\$	\$	\$	\$	\$	\$	\$	\$

## Example 0: Compute transport plan (no toolbox)

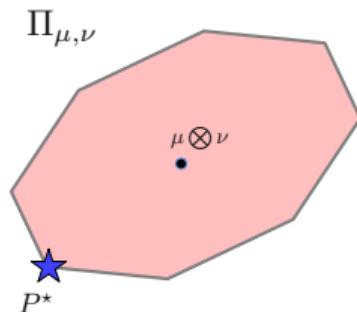
Computing the OT requires solving the linear problem:

$$\text{OT}(\mu, \nu) = \inf_{P \in \Pi_{\mu, \nu}} \sum_{i,j}^{n,m} C_{ij} P_{ij},$$

subject to

$$0 \leq P_{ij} \leq 1, \sum_{i=1}^m P_{ij} = \nu_j, \sum_{j=1}^n P_{ij} = \mu_i.$$

Feasible set:



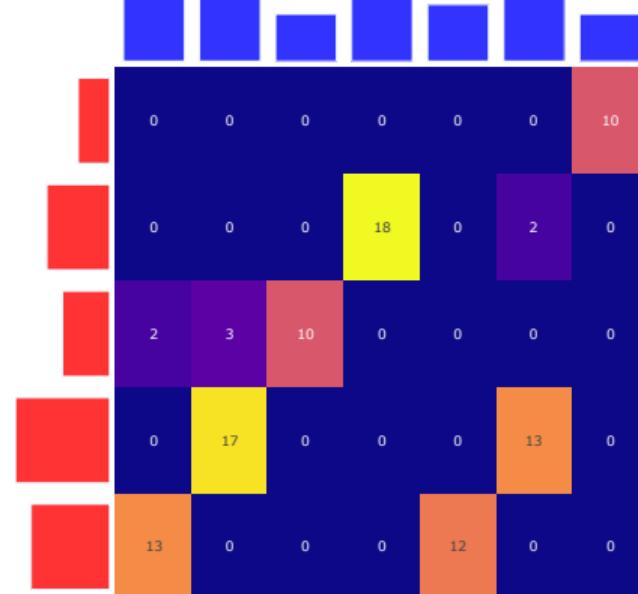
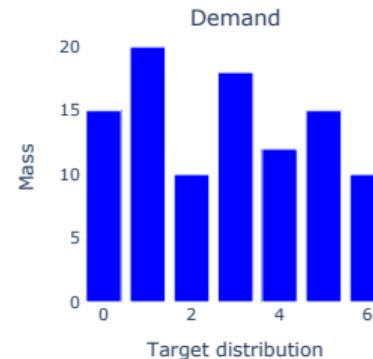
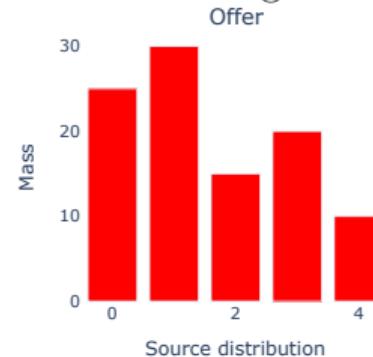
Algorithm:

- Input:  $\mu$  and  $\nu$  with supports  $s_\mu$  and  $s_\nu$
- Check distributions are valid
- Compute  $C = c(s_\mu, s_\nu)$
- Soln:  $P^* = \arg \min \sum_{i,j}^{n,m} C_{ij} P_{ij}$
- Overall cost:  $d = \sum_{i,j}^{n,m} C_{ij} P_{ij}^*$

[Notebook: Wasserstein\\_linprog.ipynb](#)

# Example 1: Discrete Kantorovich plan

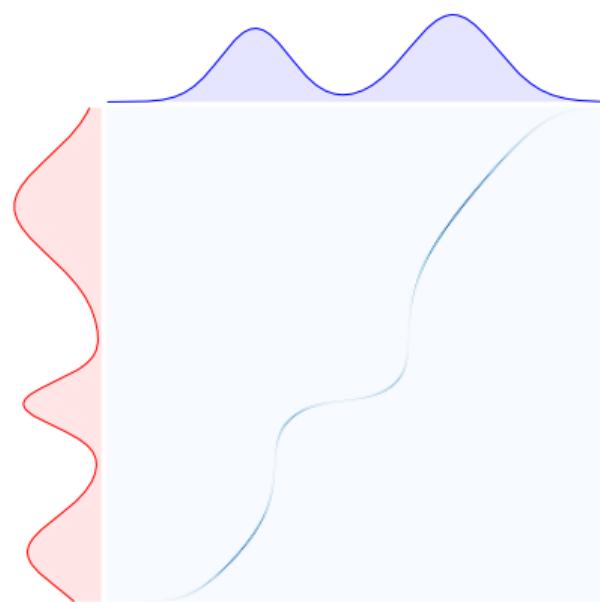
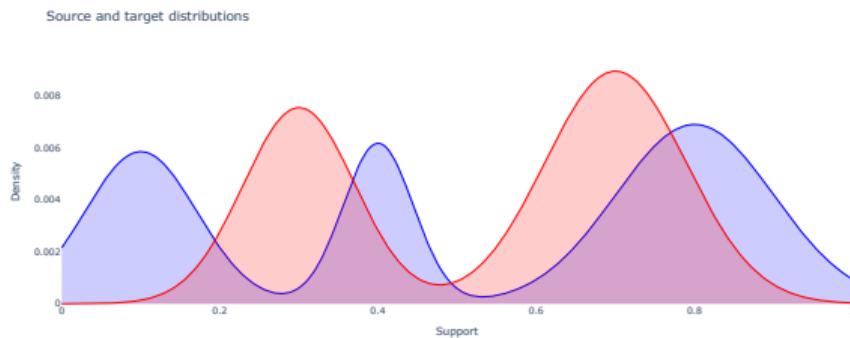
Let consider the following source and target distributions



Notebook: [kantorovich.ipynb](#)

## Example 2: Continuous Kantorovich plan

Let us now consider two distributions over a continuous support



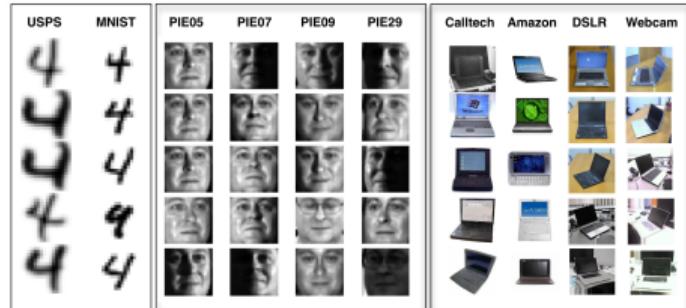
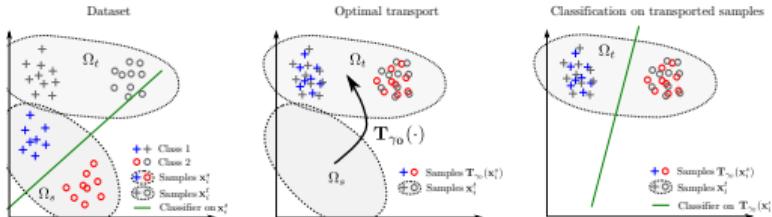
Observe that the plan remained *sparse*, i.e., the mass did not spread much

(bear this in mind, explanation to follow)

[Notebook: kantorovich.ipynb](#)

# Motivation: Domain adaptation

Same tasks, different domains



IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. X, NO. X, JANUARY XX

1

## Optimal Transport for Domain Adaptation

Nicolas Courty, Rémi Flamary, Devis Tuia, Senior Member, IEEE,  
Alain Rakotomamonjy, Member, IEEE

**Abstract**—Domain adaptation is one of the most challenging tasks of modern data analytics. If the adaptation is done correctly, models built on a specific data representation become more robust when confronted to data depicting the same classes, but described by another observation system. Among the many strategies proposed, finding domain-invariant representations has shown excellent properties, in particular since it allows to train a unique classifier effective in all domains. In this paper, we propose a regularized unsupervised optimal transportation model to perform the alignment of the representations in the source and target domains. We learn a transportation plan matching both PDFs, which constrains labeled samples of the same class in the source domain to remain close during transport. This way, we exploit at the same time the labeled samples in the source and the distributions observed in both domains. Experiments on toy and challenging real visual adaptation examples show the interest of the method, that consistently outperforms state of the art approaches. In addition, numerical experiments show that our approach leads to better performances on domain invariant deep learning features and can be easily adapted to the semi-supervised case where few labeled samples are available in the target domain.

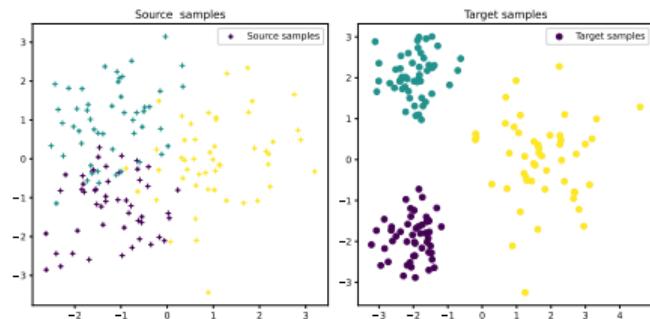
in 2016

Nicolas Courty, Rémi Flamary, Devis Tuia, Alain Rakotomamonjy. Optimal Transport for Domain Adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2016, 39 (9), pp.1853- 1865.

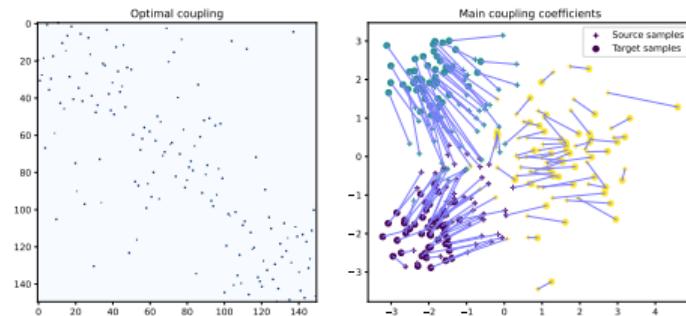
[Link: arxiv.org/abs/1507.00504](https://arxiv.org/abs/1507.00504)

# Example 3: Domain adaptation

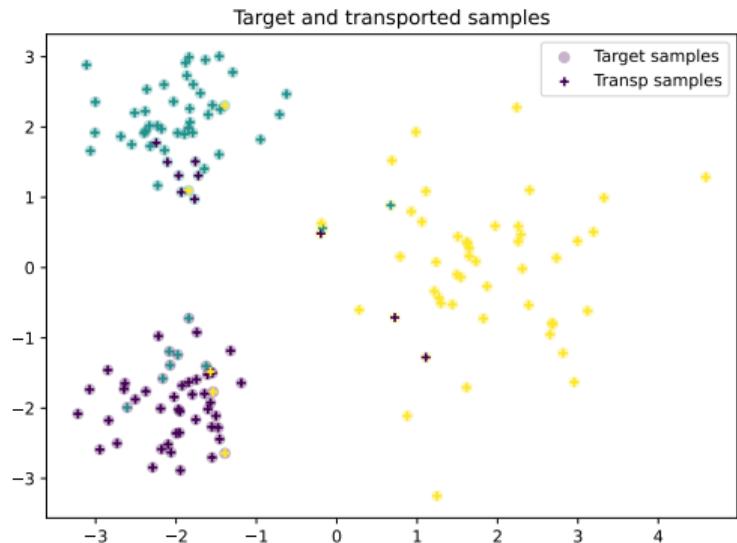
Two datasets: same task



Transport



Domain: adapted



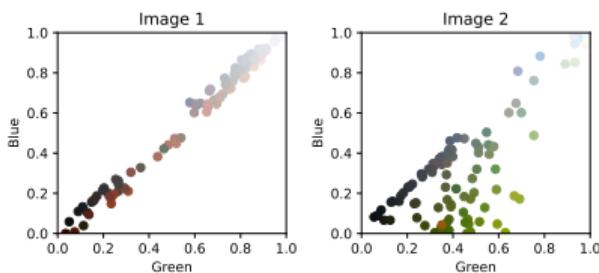
Notebook: [Domain\\_adaptation.ipynb](#)

# Example 4: Colour transfer

Original images



Histograms



[Notebook: Colour\\_transfer.ipynb](#)

Transported images

Image 1

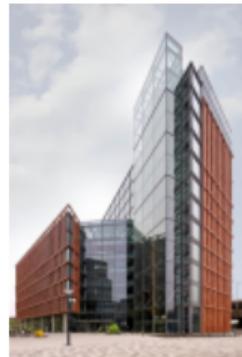


Image 1 (transported)

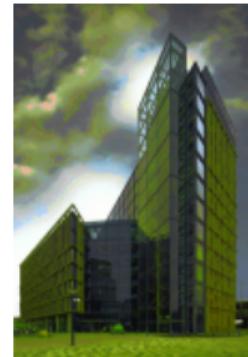


Image 2



Image 2 (transported)



## Remarks

- We've been using the cost  $c(x, y) = |x - y|^p, p \geq 1$ . In this case, if  $\mu$  and  $\nu$  are absolutely continuous wrt the Lebesgue measure, the Kantorovich problem has a unique solution. Furthermore, this solution coincides with that of Monge's problem.
- If  $p = 2$ , the optimal map is the gradient of a convex function, thus *preserving orientation*.
- In some cases, the optimal plan will require to mass splitting (e.g., in the case of atomic measures) and thus Monge's solution may fail to exist.
- From a (Kantorovich) transport plan we can always extract a transport map, e.g., via the barycentric projection

# Solving discrete OT

Let  $\mu = \sum_{i=1}^n \delta_{x_i}$  and  $\nu = \sum_{j=1}^m \delta_{y_j}$

$$\pi^* \in \arg \min_{\pi \in \Pi(\mu, \nu)} \langle C, \pi \rangle$$

- Linear problem: it can be rewritten in a vectorial form  $\min_{t \geq 0} F(t) = c^T t$
- Linear constraints of the form  $\pi \mathbb{1}_m = \mu$  and  $\pi^T \mathbb{1}_n = \nu$

$\implies$  Linear problem + linear constraints: complexity is  $\mathcal{O}(n^3 \log(n))$

$\implies$  Need for solvers that provide approximate solutions! See [Peyré et Cuturi 2019]

# Regularization of OT

$$\pi_\varepsilon = \arg \min_{\pi \in \mathbb{R}_+^{n \times m}} \langle C, \pi \rangle + \varepsilon \Omega(\pi)$$

Advantages of regularizing the optimisation problem:

- Fast and *smoother* solutions
- Encoding task's prior knowledge
- Better posed problem (convexity, stability).

Regularisation functionals

- Entropic regularization [Cuturi, 2013]
- KL, Itakura Saito,  $\beta$ -divergences, [Papadakis & Papadakis, 2018]

# Entropy regularized OT [Cuturi, 2013]

The solution of

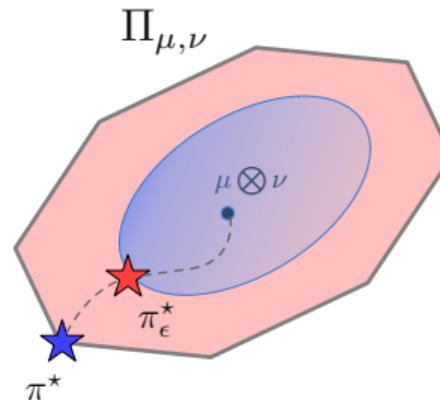
$$\text{OT}(\mu, \nu) = \min_{\pi \in \Pi(a, b)} \langle C, \pi \rangle + \varepsilon \sum_{i,j} \pi_{ij} \log(\pi_{ij})$$

is of the form

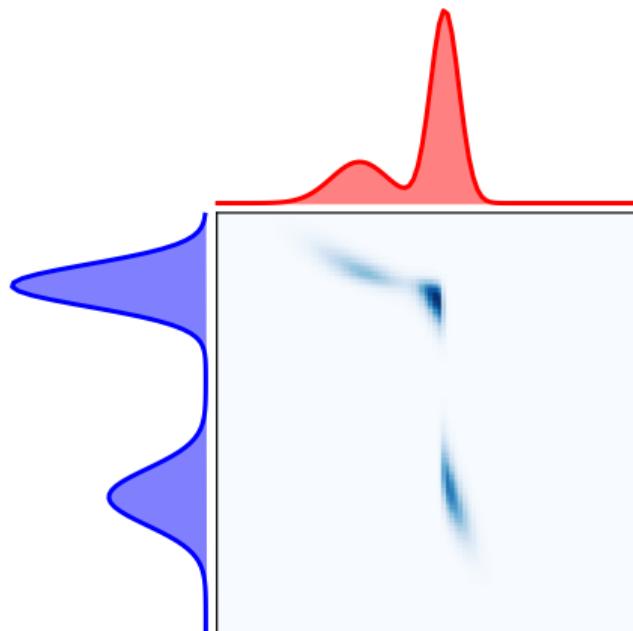
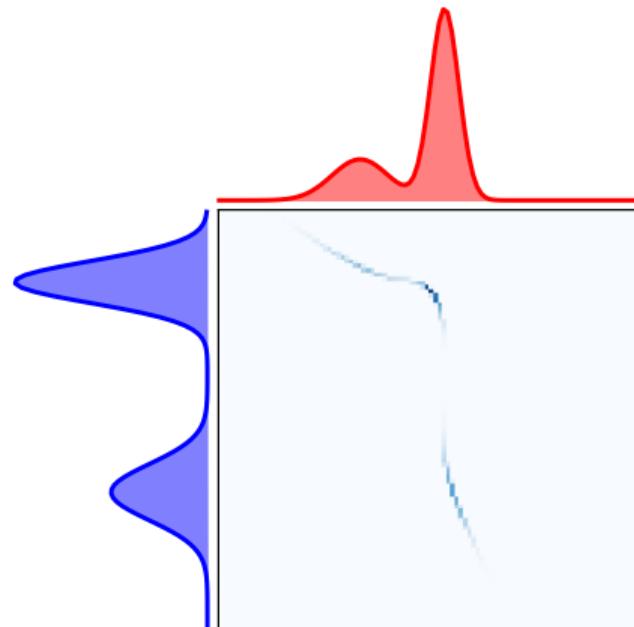
$$\pi_\varepsilon^* = \text{diag}(u) \exp \left( -\frac{C}{\varepsilon} \right) \text{diag}(v)$$

where  $u$  and  $v$  follow an alternating recursive formula.

- From Sinkhorn theorem [Sinkhorn, 1964], we know that  $\text{diag}(u)$  and  $\text{diag}(v)$  exist and are unique.
- Sinkhorn-Knopp algorithm [Knight, 2008] allows to solve it efficiently



## Example 5: Sinkhorn regularisation and effect



[Notebook: sinkhorn.ipynb](#)

## Part II

# The Wasserstein distance and metric properties

# OT lifts a distance from the measures' support

A family of distances between measures

The Kantorovitch problem

$$P^* \in \inf_{P \in \Pi_{\mu, \nu}} \langle P, C \rangle = \sum_{i,j}^{n,m} C_{ij} P_{ij}$$

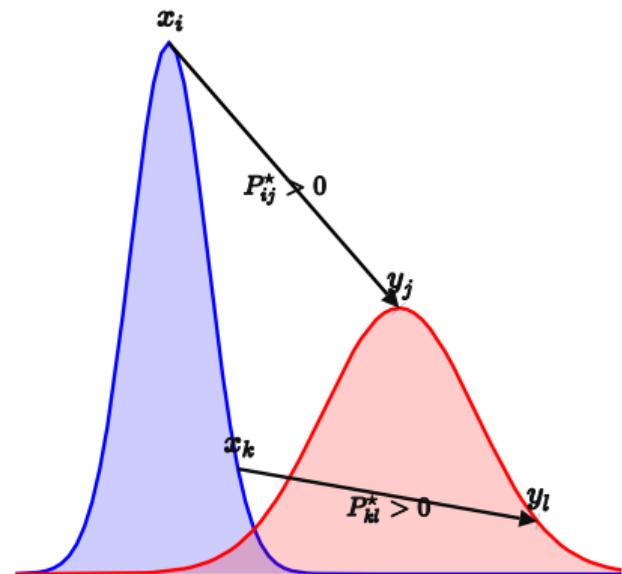
allows defining the **Wasserstein distance** of order  $p$

$$W_p^p(\mu, \nu) = \langle P^*, C \rangle$$

where the moving cost  $c(x, y) = d(x, y)^p = \|x - y\|^p$ .

It is often depicted as an “horizontal” distance

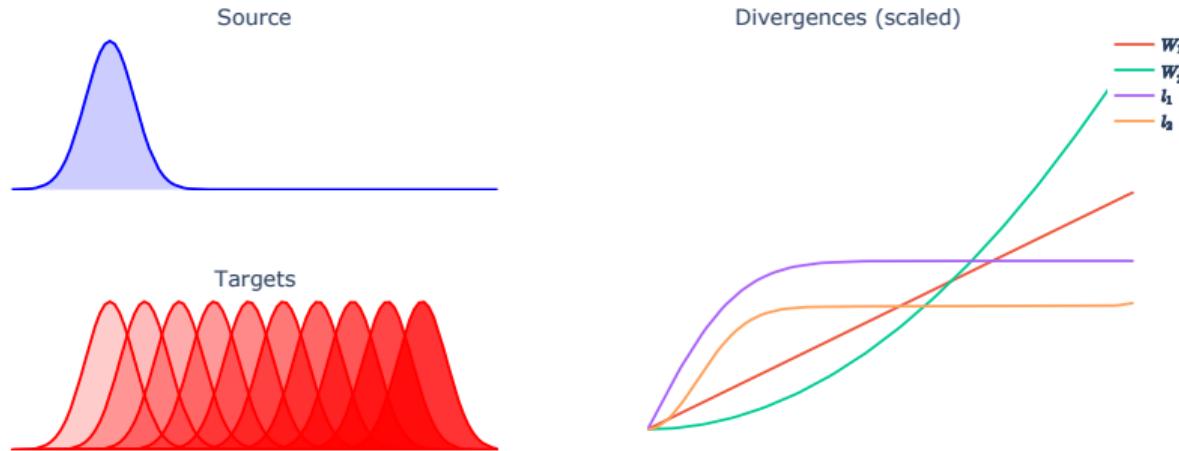
- ✓ symmetry
- ✓ identity of indiscernibles
- ✓ triangular inequality



Notebook: Horizontal distance.ipynb

# The Wasserstein distance (vs others)

- Does not need overlapping support (as KL)
- Determines the *degree of dissimilarity* between distributions



[Notebook: Wasserstein\\_distance.ipynb](#)

# On the suitability of $W_p$ for learning

- Thus far, we have referred to *spaces of probability functions*, but we are interested in applying  $W_p$  on spaces of **generative models**.
- Learning in such a space requires, more than a distance, a notion of **convergence**
- Consider  $\mu_{\text{data}}$  to be the true data distribution. We want to find a model  $(P_\theta)_{\theta \in \Theta}$  such that  $P_\theta \rightarrow \mu_{\text{data}}$ , or equivalently,  $D(\mu_{\text{data}}, P_\theta) \rightarrow 0$  — for a **reasonable** divergence  $D$ .

**Discussion:** Consider  $\delta_{x_0}$  and  $\delta_{x_i}, x_i \rightarrow x_0$

OT allows for **assessing** convergence and **constructing** convergent sequences

## Example 6: Gradient flows on Wasserstein space

Wasserstein space  $\mathbb{W}_p$ : space endowed with the distance  $W_p$

- In the space  $\mathbb{W}_p(\mathbb{R}^d)$ , we have  $W_p(\mu_n, \mu) \rightarrow 0$  iff  $\mu_n \rightarrow \mu$  (weak topology)

Consider the loss  $W_2^2(\mu_t, \mu)$ . The figure below shows how a distribution  $\mu_0$  evolves under de application of gradient flow of this loss.



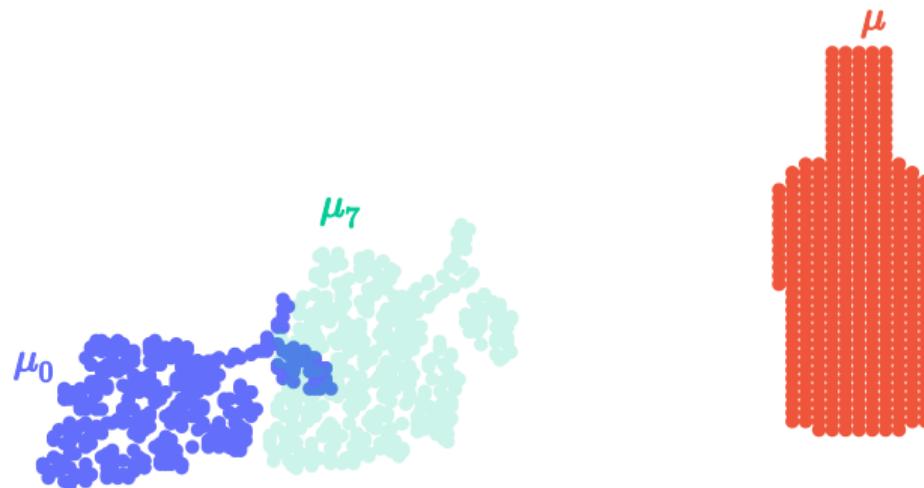
Notebook: Wasserstein Gradient Flows.ipynb

## Example 6: Gradient flows on Wasserstein space

Wasserstein space  $\mathbb{W}_p$ : space endowed with the distance  $W_p$

- In the space  $\mathbb{W}_p(\mathbb{R}^d)$ , we have  $W_p(\mu_n, \mu) \rightarrow 0$  iff  $\mu_n \rightarrow \mu$  (weak topology)

Consider the loss  $W_2^2(\mu_t, \mu)$ . The figure below shows how a distribution  $\mu_0$  evolves under de application of gradient flow of this loss.



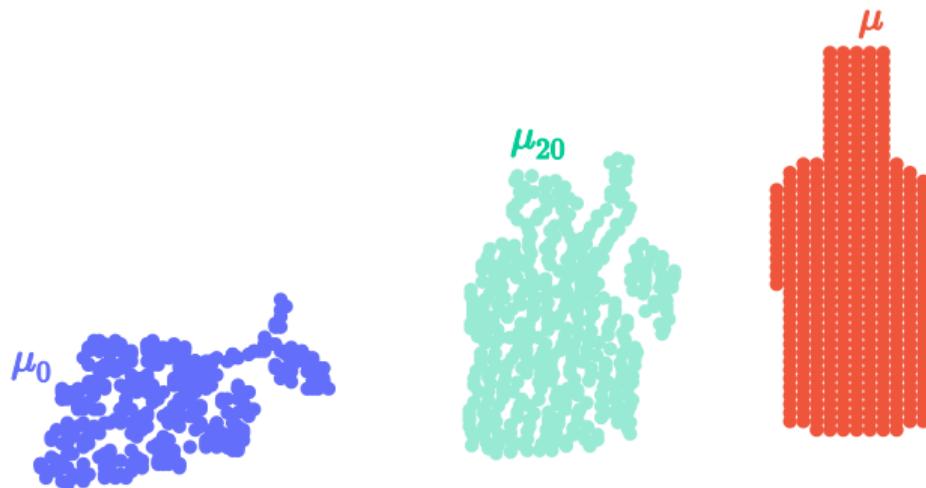
Notebook: Wasserstein Gradient Flows.ipynb

## Example 6: Gradient flows on Wasserstein space

Wasserstein space  $\mathbb{W}_p$ : space endowed with the distance  $W_p$

- In the space  $\mathbb{W}_p(\mathbb{R}^d)$ , we have  $W_p(\mu_n, \mu) \rightarrow 0$  iff  $\mu_n \rightarrow \mu$  (weak topology)

Consider the loss  $W_2^2(\mu_t, \mu)$ . The figure below shows how a distribution  $\mu_0$  evolves under de application of gradient flow of this loss.



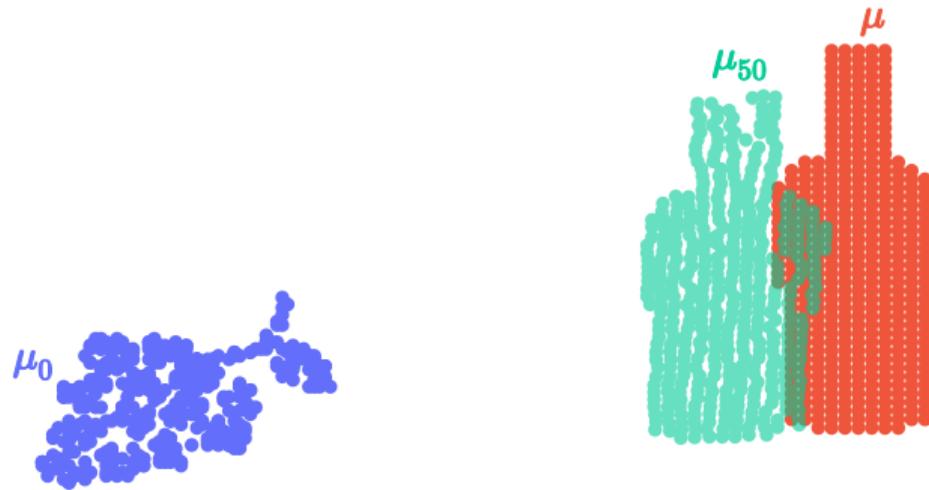
Notebook: Wasserstein Gradient Flows.ipynb

## Example 6: Gradient flows on Wasserstein space

Wasserstein space  $\mathbb{W}_p$ : space endowed with the distance  $W_p$

- In the space  $\mathbb{W}_p(\mathbb{R}^d)$ , we have  $W_p(\mu_n, \mu) \rightarrow 0$  iff  $\mu_n \rightarrow \mu$  (weak topology)

Consider the loss  $W_2^2(\mu_t, \mu)$ . The figure below shows how a distribution  $\mu_0$  evolves under de application of gradient flow of this loss.



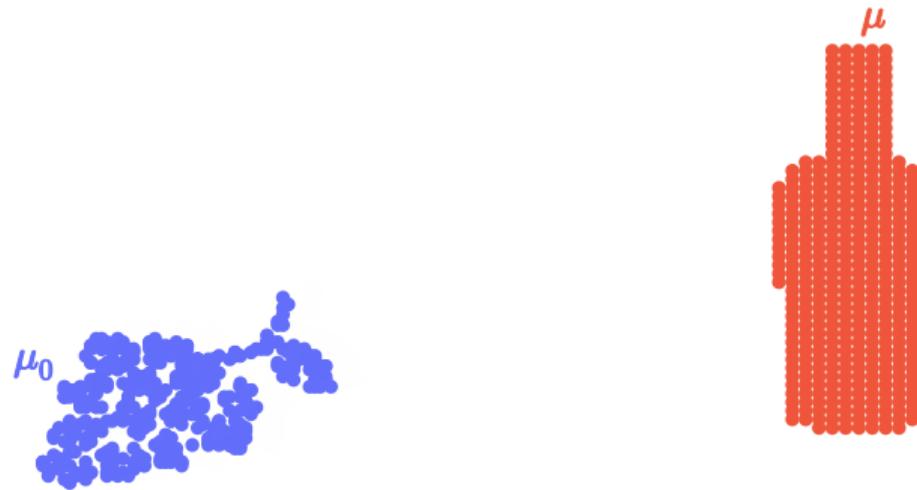
Notebook: Wasserstein Gradient Flows.ipynb

## Example 6: Gradient flows on Wasserstein space

Wasserstein space  $\mathbb{W}_p$ : space endowed with the distance  $W_p$

- In the space  $\mathbb{W}_p(\mathbb{R}^d)$ , we have  $W_p(\mu_n, \mu) \rightarrow 0$  iff  $\mu_n \rightarrow \mu$  (weak topology)

Consider the loss  $W_2^2(\mu_t, \mu)$ . The figure below shows how a distribution  $\mu_0$  evolves under de application of gradient flow of this loss.



Notebook: Wasserstein Gradient Flows.ipynb

# Geodesic paths between distributions

A geodesic generalizes the concept of a straight line between two points

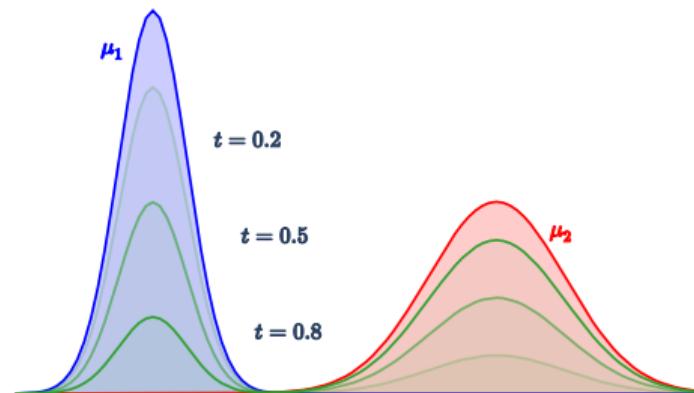


It is a curve that represents the shortest path between two manifolds

Euclidean space with a  $l_2$  distance is a **geodesic space**

$$\forall t \in [0, 1], \quad \mu^{1 \rightarrow 2}(t) = t\mu_2 + (1 - t)\mu_1$$

Allows “vertical” interpolation between the distributions



[Notebook: Wasserstein Geodesics.ipynb](#)

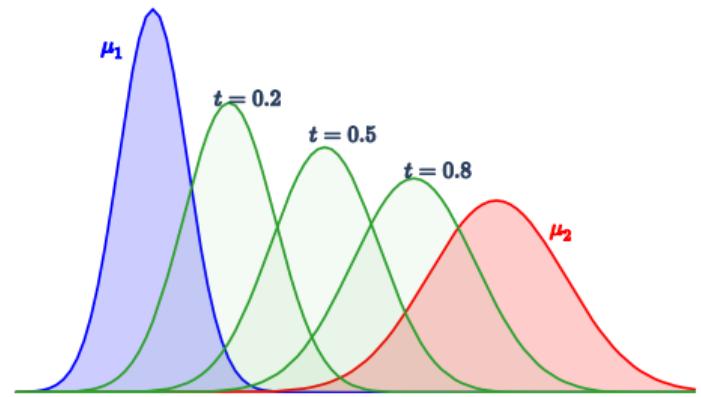
# Geodesic properties of the Wasserstein space

$\mathbb{W}_p$  is a **geodesic space**

- Given a Monge map  $T$  between  $\mu_1$  and  $\mu_2$  such that  $T_{\#}\mu_1 = \mu_2$ , a geodesic curve  $\mu^{1 \rightarrow 2}$  is

$$\forall t \in [0, 1], \quad \mu^{1 \rightarrow 2}(t) = (tT + (1-t)\text{Id})_{\#}\mu_1$$

- It represents the shortest path (on the Wasserstein space  $\mathbb{W}_p$ ) between  $\mu_1$  and  $\mu_2$
- Allows “horizontal” interpolation between the distributions



**Notebook:** [Wasserstein Geodesics.ipynb](#)

# The Wasserstein barycentre

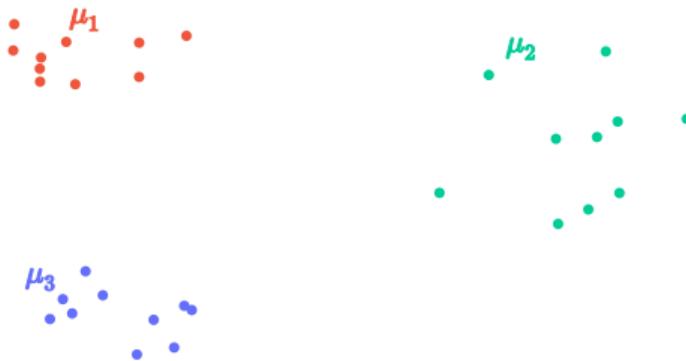
Given a set of distributions  $\mu_s$ , compute:

$$\bar{\mu} = \arg \min_{\mu} \sum_{i=1}^s \lambda_i W_p^p(\mu, \mu_i)$$

where  $\lambda_i > 0$  and  $\sum_{i=1}^s \lambda_i = 1$ .

Generalizes the interpolation between more than 2 measures.

For discrete measures  $\mu = \sum_{i=1}^n a_i \delta_{x_i} \Rightarrow$  we can fix the weights  $a_i$  and/or the support  $x_i$ .



[Notebook: Wasserstein barycenter.ipynb](#)

# The Wasserstein barycentre

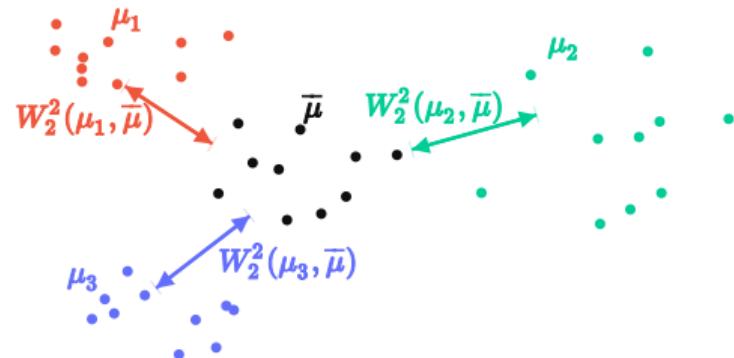
Given a set of distributions  $\mu_s$ , compute:

$$\bar{\mu} = \arg \min_{\mu} \sum_{i=1}^s \lambda_i W_p^p(\mu, \mu_i)$$

where  $\lambda_i > 0$  and  $\sum_{i=1}^s \lambda_i = 1$ .

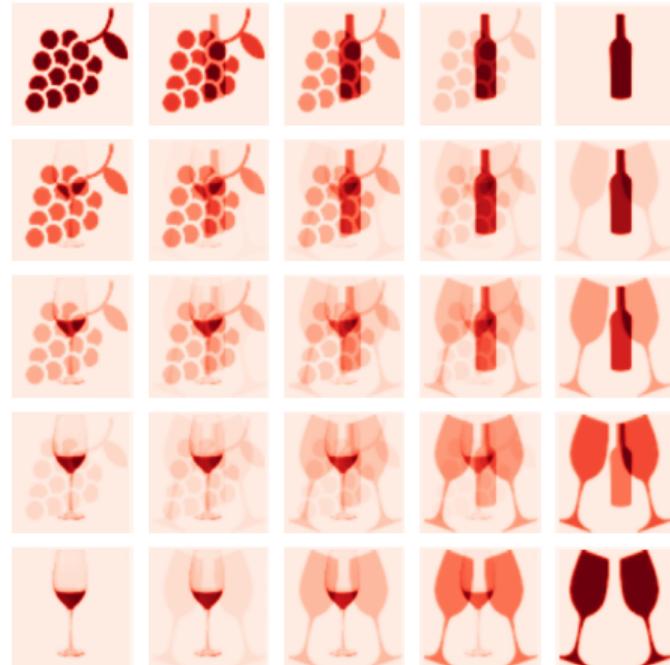
Generalizes the interpolation between more than 2 measures.

For discrete measures  $\mu = \sum_{i=1}^n a_i \delta_{x_i} \Rightarrow$  we can fix the weights  $a_i$  and/or the support  $x_i$ .



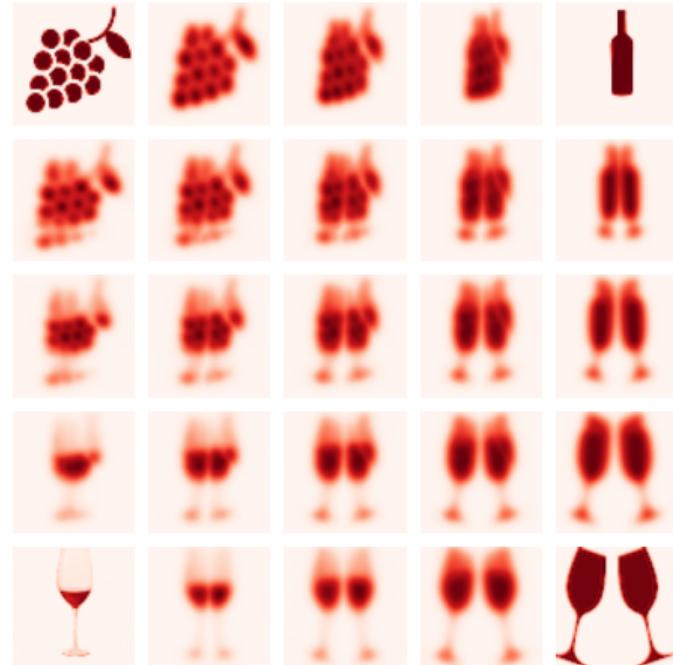
# The Wasserstein barycentre

Example on averaging over images



**Figure 1:** In the Euclidean space

Notebook: Wass bary 4 distribs.ipynb



**Figure 2:** In the Wasserstein space

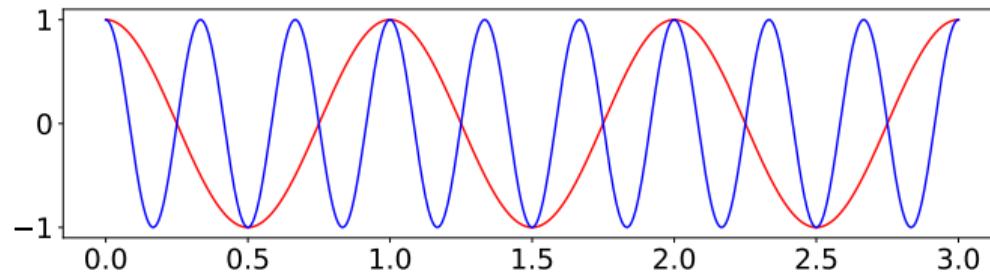
(if time allows)

## Part III

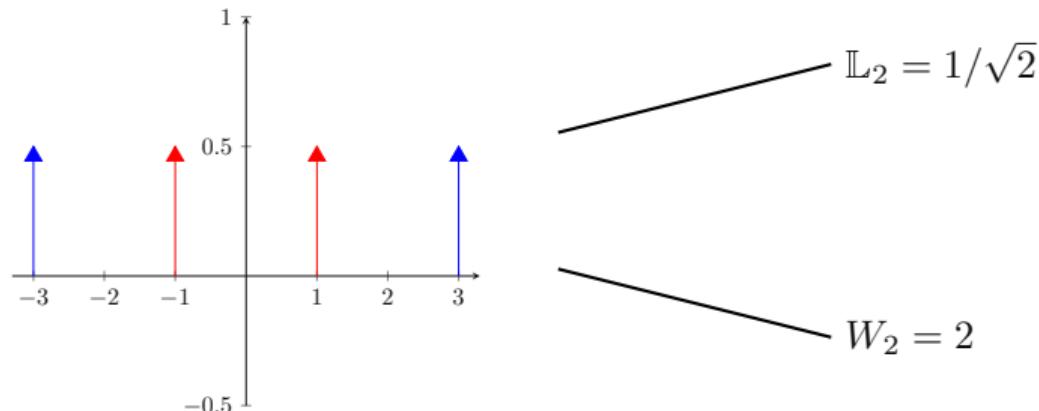
### OT for time series: The Wasserstein-Fourier distance

# Applying the Wasserstein distance to time series

Two cosine signals with frequencies 1 and 3.



The associated PSD functions .



# Definition: The Wasserstein-Fourier distance

## Definition

For two signals  $x$  and  $y$  belonging to two different classes of time series, we denote by

- $[x]$  and  $[y]$  their respective class
- $s_x$  and  $s_y$  their respective normalised PSD (NPSD)

We define the proposed *Wasserstein-Fourier* (WF) distance:

$$\text{WF}([x], [y]) = W_2(s_x, s_y).$$

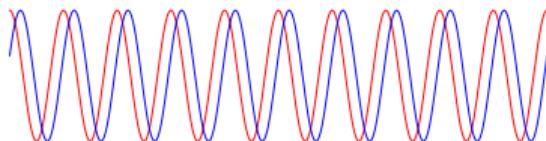
## Theorem

*WF is a distance over the space of equivalence classes of time series sharing the same NPSD.*

E. Cazelles, A. Robert & **F. Tobar**, The Wasserstein-Fourier Distance for Stationary Time Series. *IEEE Trans. on Signal Processing* 2021.

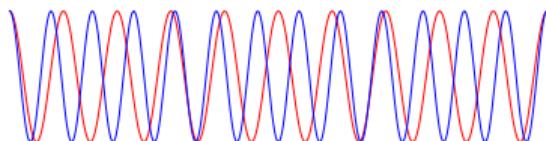
# Basics properties of the WF distance

Time shifting :  $x(t) = y(t - t_0)$ .



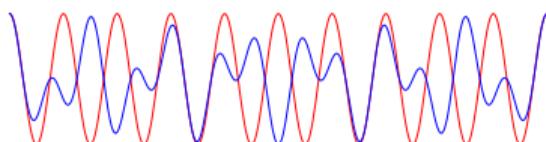
$$\text{WF}([x], [y]) = 0$$

Time scaling :  $x(t) = y(at), a > 0$ .



$$\text{WF}([x], [y]) = |a - 1|(\langle |Y|^2 \rangle_{s_y})^{\frac{1}{2}}$$

Frequency shifting :  $x(t) = e^{2i\pi\xi_0 t}y(t)$ .



$$\text{WF}([x], [y]) = |\xi_0|$$

# How to interpolate two time series?

**The usual  $\mathbb{L}_2$  path:** a superposition of two signals

$$x_\gamma(t) = \gamma \textcolor{red}{x_1(t)} + (1 - \gamma) \textcolor{blue}{x_2(t)}, \quad \gamma \in [0, 1],$$

**Example:** For EEG, the  $\mathbb{L}_2$  average of multiple responses to a common stimulus would probably convey little information about the true average response and it is likely to quickly vanish due to the random phases.

Toy example: The VF path i.e. Wasserstein interpolation in the frequency domain



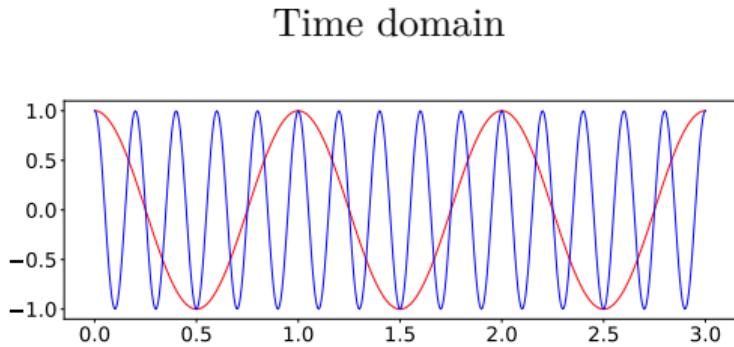
# How to interpolate two time series?

**The usual  $\mathbb{L}_2$  path:** a superposition of two signals

$$x_\gamma(t) = \gamma \textcolor{red}{x_1(t)} + (1 - \gamma) \textcolor{blue}{x_2(t)}, \quad \gamma \in [0, 1],$$

**Example:** For EEG, the  $\mathbb{L}_2$  average of multiple responses to a common stimulus would probably convey little information about the true average response and it is likely to quickly vanish due to the random phases.

**Toy example: The WF path i.e. Wasserstein interpolation in the frequency domain**



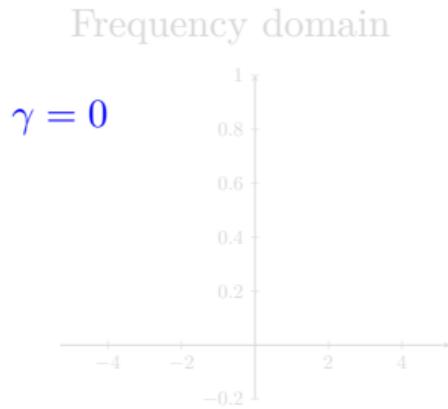
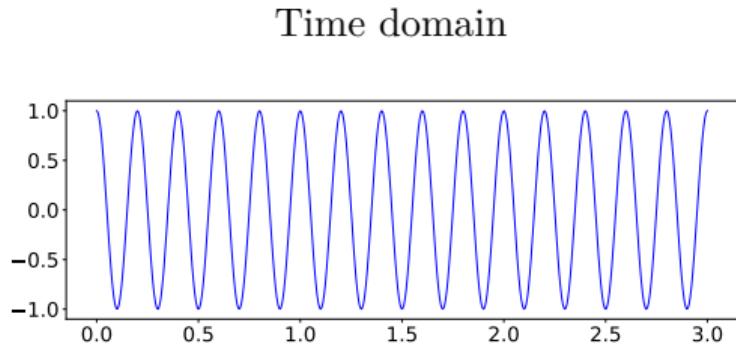
# How to interpolate two time series?

The usual  $\mathbb{L}_2$  path: a superposition of two signals

$$x_\gamma(t) = \gamma \textcolor{red}{x_1(t)} + (1 - \gamma) \textcolor{blue}{x_2(t)}, \quad \gamma \in [0, 1],$$

**Example:** For EEG, the  $\mathbb{L}_2$  average of multiple responses to a common stimulus would probably convey little information about the true average response and it is likely to quickly vanish due to the random phases.

**Toy example: The WF path i.e. Wasserstein interpolation in the frequency domain**



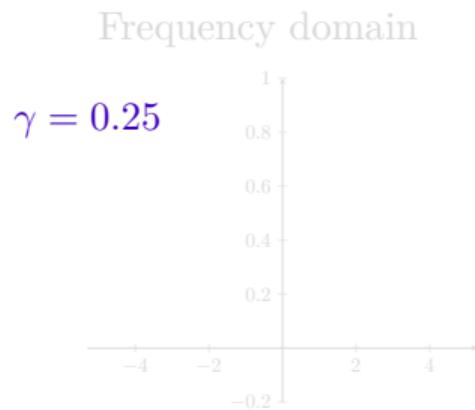
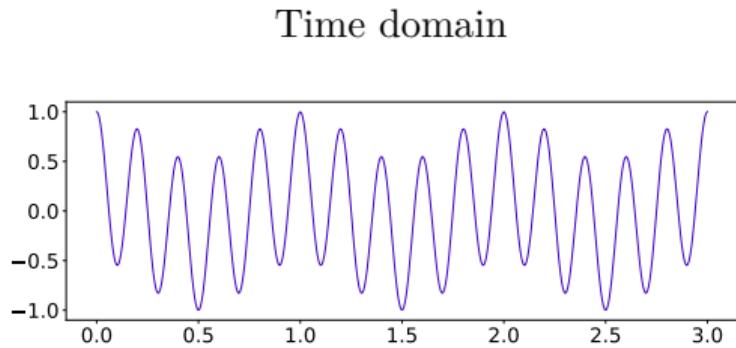
# How to interpolate two time series?

The usual  $\mathbb{L}_2$  path: a superposition of two signals

$$x_\gamma(t) = \gamma \textcolor{red}{x_1(t)} + (1 - \gamma) \textcolor{blue}{x_2(t)}, \quad \gamma \in [0, 1],$$

**Example:** For EEG, the  $\mathbb{L}_2$  average of multiple responses to a common stimulus would probably convey little information about the true average response and it is likely to quickly vanish due to the random phases.

**Toy example: The WF path i.e. Wasserstein interpolation in the frequency domain**



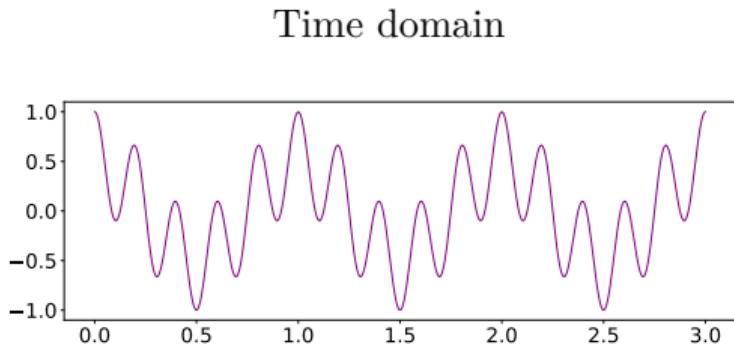
# How to interpolate two time series?

**The usual  $\mathbb{L}_2$  path:** a superposition of two signals

$$x_\gamma(t) = \gamma \textcolor{red}{x_1(t)} + (1 - \gamma) \textcolor{blue}{x_2(t)}, \quad \gamma \in [0, 1],$$

**Example:** For EEG, the  $\mathbb{L}_2$  average of multiple responses to a common stimulus would probably convey little information about the true average response and it is likely to quickly vanish due to the random phases.

**Toy example: The WF path i.e. Wasserstein interpolation in the frequency domain**



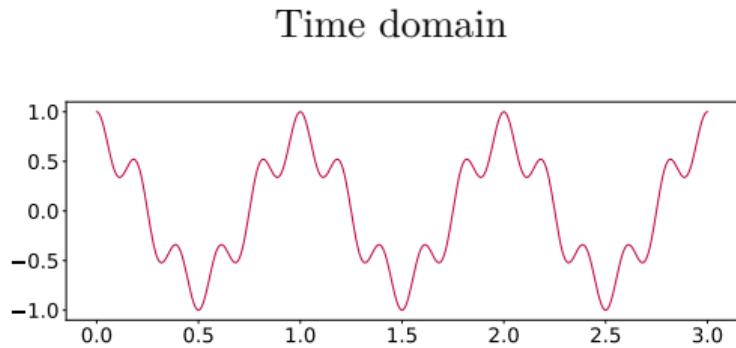
# How to interpolate two time series?

The usual  $\mathbb{L}_2$  path: a superposition of two signals

$$x_\gamma(t) = \gamma \textcolor{red}{x_1(t)} + (1 - \gamma) \textcolor{blue}{x_2(t)}, \quad \gamma \in [0, 1],$$

**Example:** For EEG, the  $\mathbb{L}_2$  average of multiple responses to a common stimulus would probably convey little information about the true average response and it is likely to quickly vanish due to the random phases.

**Toy example: The WF path i.e. Wasserstein interpolation in the frequency domain**



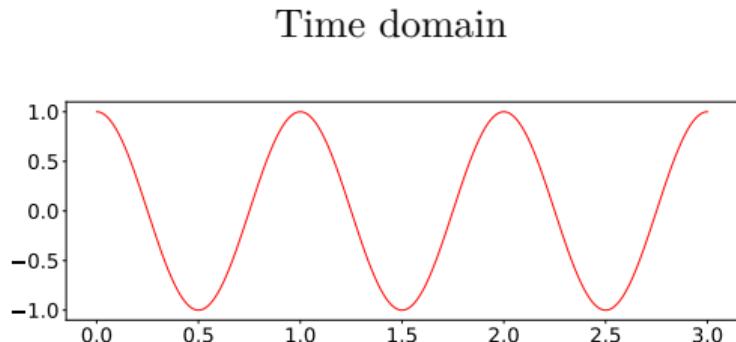
# How to interpolate two time series?

The usual  $\mathbb{L}_2$  path: a superposition of two signals

$$x_\gamma(t) = \gamma \textcolor{red}{x_1(t)} + (1 - \gamma) \textcolor{blue}{x_2(t)}, \quad \gamma \in [0, 1],$$

**Example:** For EEG, the  $\mathbb{L}_2$  average of multiple responses to a common stimulus would probably convey little information about the true average response and it is likely to quickly vanish due to the random phases.

**Toy example: The WF path i.e. Wasserstein interpolation in the frequency domain**



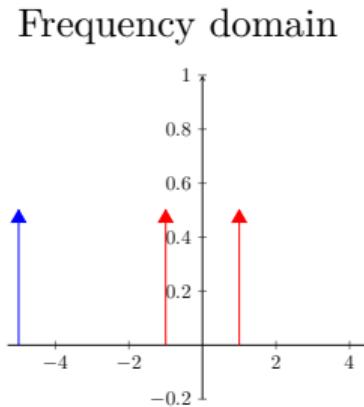
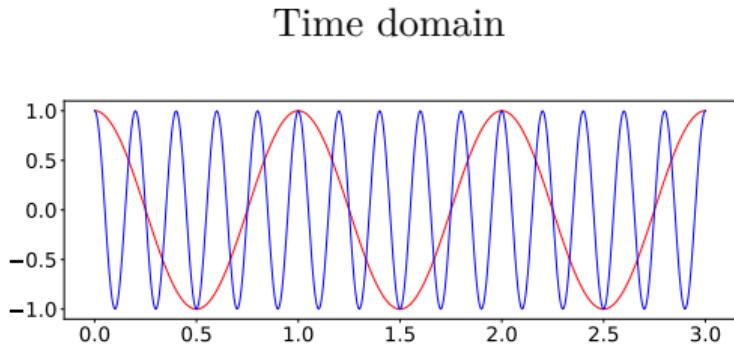
# How to interpolate two time series?

The usual  $\mathbb{L}_2$  path: a superposition of two signals

$$x_\gamma(t) = \gamma \textcolor{red}{x_1(t)} + (1 - \gamma) \textcolor{blue}{x_2(t)}, \quad \gamma \in [0, 1],$$

**Example:** For EEG, the  $\mathbb{L}_2$  average of multiple responses to a common stimulus would probably convey little information about the true average response and it is likely to quickly vanish due to the random phases.

**Toy example: The WF path i.e. Wasserstein interpolation in the frequency domain**



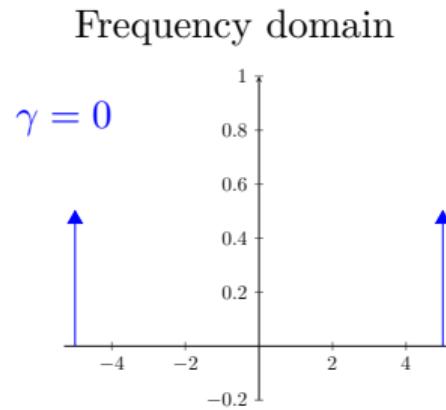
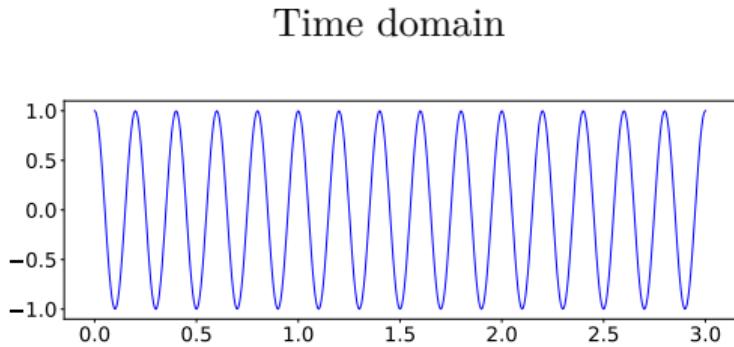
# How to interpolate two time series?

The usual  $\mathbb{L}_2$  path: a superposition of two signals

$$x_\gamma(t) = \gamma \textcolor{red}{x_1(t)} + (1 - \gamma) \textcolor{blue}{x_2(t)}, \quad \gamma \in [0, 1],$$

**Example:** For EEG, the  $\mathbb{L}_2$  average of multiple responses to a common stimulus would probably convey little information about the true average response and it is likely to quickly vanish due to the random phases.

**Toy example: The WF path i.e. Wasserstein interpolation in the frequency domain**



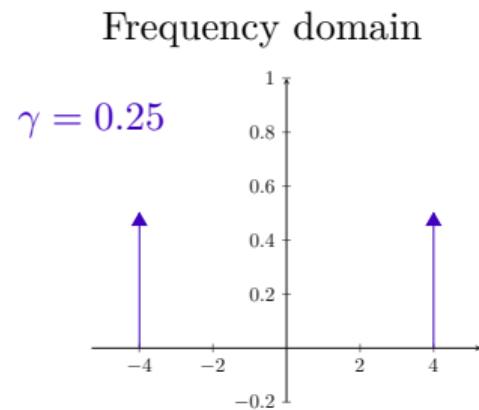
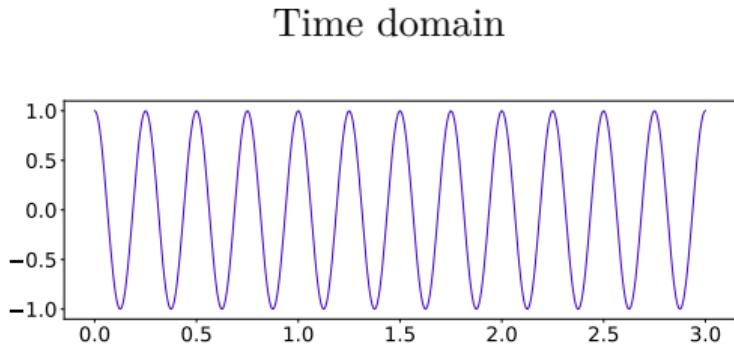
# How to interpolate two time series?

The usual  $\mathbb{L}_2$  path: a superposition of two signals

$$x_\gamma(t) = \gamma \textcolor{red}{x_1(t)} + (1 - \gamma) \textcolor{blue}{x_2(t)}, \quad \gamma \in [0, 1],$$

**Example:** For EEG, the  $\mathbb{L}_2$  average of multiple responses to a common stimulus would probably convey little information about the true average response and it is likely to quickly vanish due to the random phases.

**Toy example: The WF path i.e. Wasserstein interpolation in the frequency domain**



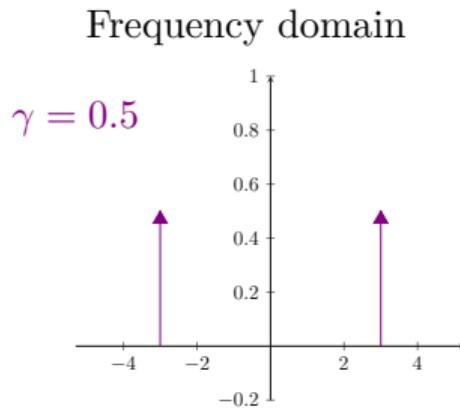
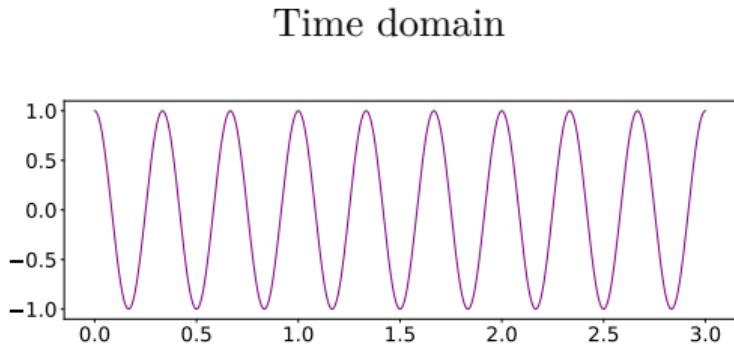
# How to interpolate two time series?

The usual  $\mathbb{L}_2$  path: a superposition of two signals

$$x_\gamma(t) = \gamma \textcolor{red}{x_1(t)} + (1 - \gamma) \textcolor{blue}{x_2(t)}, \quad \gamma \in [0, 1],$$

**Example:** For EEG, the  $\mathbb{L}_2$  average of multiple responses to a common stimulus would probably convey little information about the true average response and it is likely to quickly vanish due to the random phases.

**Toy example: The WF path i.e. Wasserstein interpolation in the frequency domain**



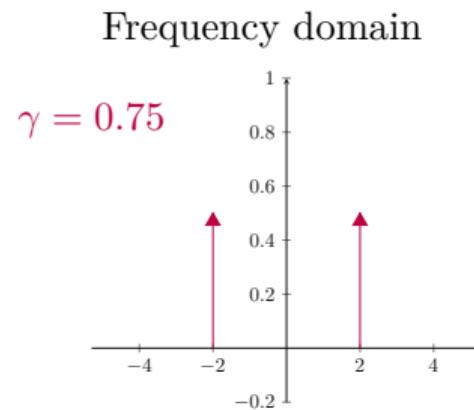
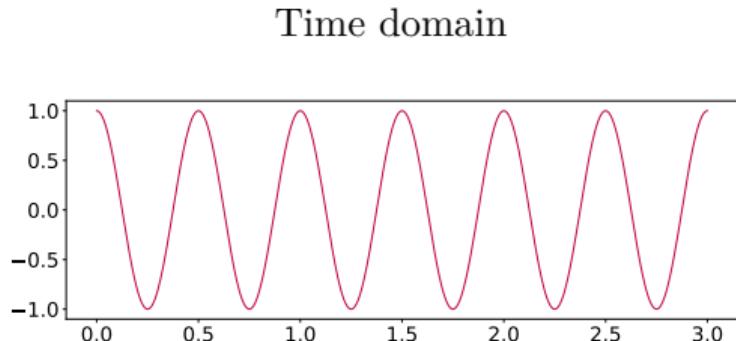
# How to interpolate two time series?

The usual  $\mathbb{L}_2$  path: a superposition of two signals

$$x_\gamma(t) = \gamma \textcolor{red}{x_1(t)} + (1 - \gamma) \textcolor{blue}{x_2(t)}, \quad \gamma \in [0, 1],$$

**Example:** For EEG, the  $\mathbb{L}_2$  average of multiple responses to a common stimulus would probably convey little information about the true average response and it is likely to quickly vanish due to the random phases.

**Toy example: The WF path i.e. Wasserstein interpolation in the frequency domain**



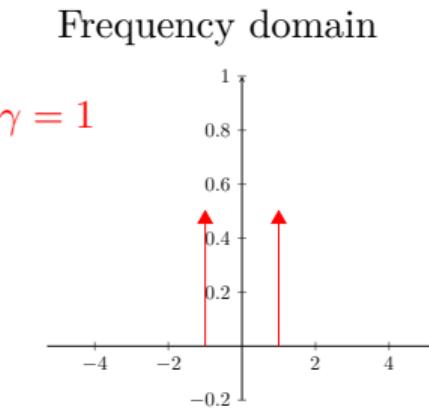
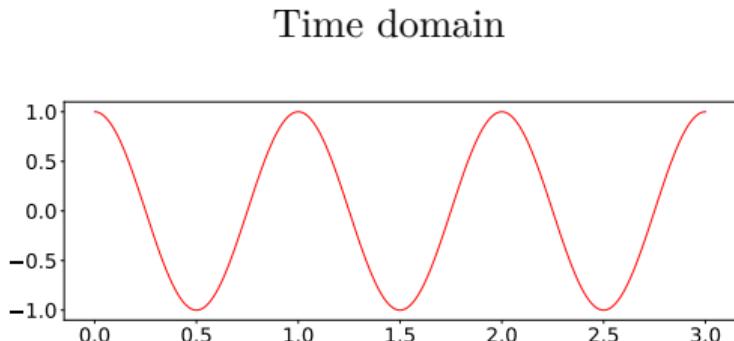
# How to interpolate two time series?

The usual  $\mathbb{L}_2$  path: a superposition of two signals

$$x_\gamma(t) = \gamma \textcolor{red}{x_1(t)} + (1 - \gamma) \textcolor{blue}{x_2(t)}, \quad \gamma \in [0, 1],$$

**Example:** For EEG, the  $\mathbb{L}_2$  average of multiple responses to a common stimulus would probably convey little information about the true average response and it is likely to quickly vanish due to the random phases.

**Toy example: The WF path i.e. Wasserstein interpolation in the frequency domain**



# An interpolation path between two times series

Time domain

Frequency domain

NPSD

$x_1, x_2$

$s_1, s_2$

McCann's interpolant (or constant-speed geodesic, Ambrosio et. al (2008))  
 $(g_\gamma)_{\gamma \in [0,1]}$  between  $s_1$  and  $s_2$ .

Inverse Fourier transform

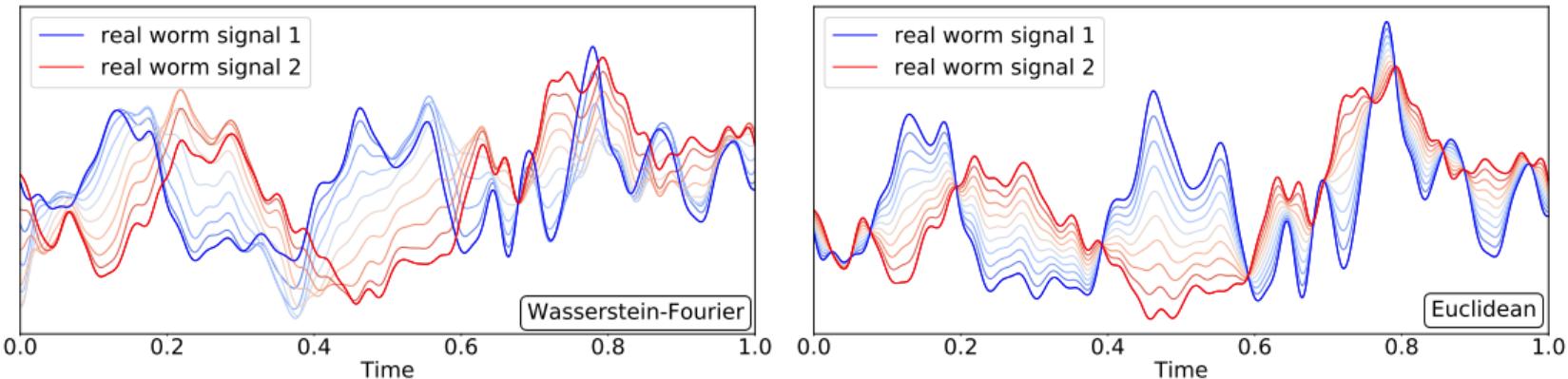
$(x_\gamma)_{\gamma \in [0,1]}$

Interpolant between  
 $x_1$  and  $x_2$

$g_\gamma = p_\gamma \# \pi^*, \gamma \in [0, 1]$

- $p_\gamma(u, v) = (1 - \gamma)u + \gamma v$ , for  $u, v \in \mathbb{R}$
- $\pi^*$  optimal transport plan between  $s_1$  and  $s_2$
- $\#$  = pushforward operator

## Example: interpolation for the *C. Elegans* database



10-step interpolation  $(x_\gamma)_{\gamma \in [0,1]}$  between two signals from the *C. elegans* database using the proposed WF distance (top) and the Euclidean distance (bottom): the true signals are shown in solid blue and red, while the interpolations are colour-coded with respect to  $\gamma$ .

# Logistic regression of time series

For two classes  $C_0$  and  $C_1$ , one defines a binary classification of a sample  $s$  as

$$p(C_0|s) = \frac{1}{1 + e^{-\alpha + \beta d(s, \bar{s}_0) + \gamma d(s, \bar{s}_1)}},$$

where  $d$  is a divergence ( $\mathbb{L}_2, KL, W_2$ ) and  $\bar{s}_i$  sums up the information of class  $C_i$ .

- $\mathbb{L}_2$  and  $KL$  cases:

$$\bar{s} \in \arg \min_s \frac{1}{n} \sum_{i=1}^n \|s_i - s\|^2 = \frac{1}{n} \sum_{i=1}^n s_i.$$

- $W_2$  case: a **Wasserstein barycentre** of a family  $(s_i)_{i=1,\dots,n}$  of distributions is given by

$$\bar{s} \in \arg \min_s \frac{1}{n} \sum_{i=1}^n W_2^2(s_i, s).$$

# Logistic regression of time series

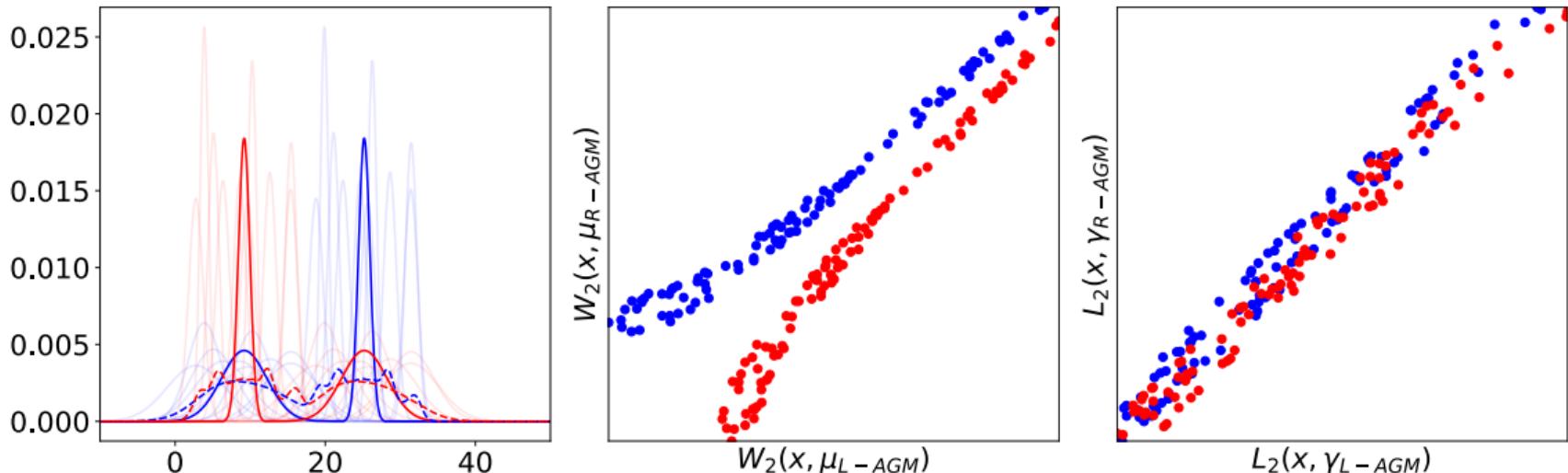


Illustration of the linear separability made possible by the Wasserstein-Fourier distance.

# Real-world example: urban audio recordings<sup>2</sup>

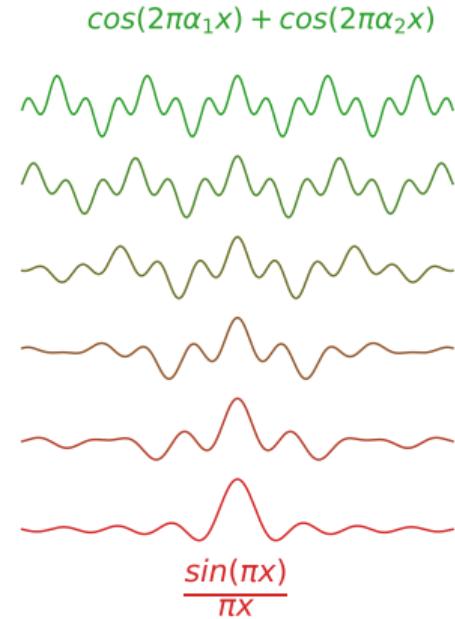
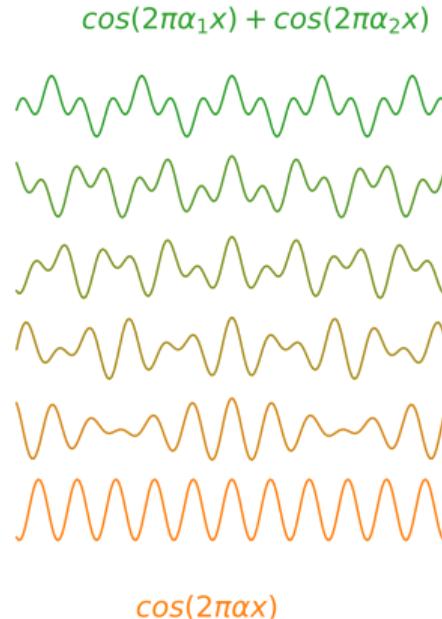
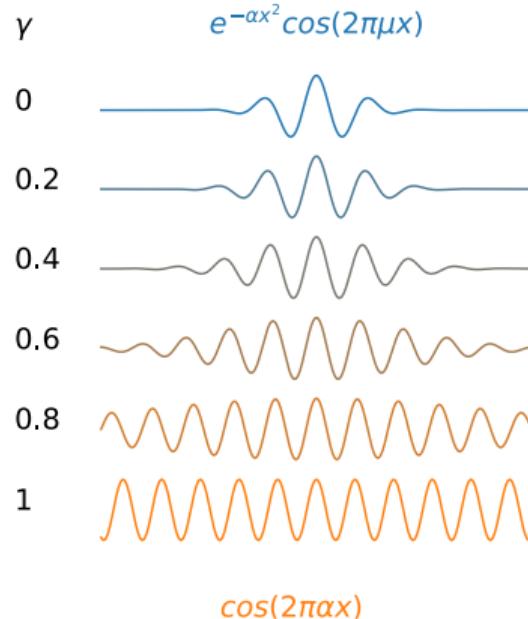
	$\mathcal{L}_{W_2}$	$\mathcal{L}_{\mathbb{L}_2}$	$\mathcal{L}_{KL}$
air conditioner	<b>0.732</b> ( $\pm 0.072$ )	0.718 ( $\pm 0.047$ )	0.650 ( $\pm 0.090$ )
car horn	0.588 ( $\pm 0.077$ )	0.743 ( $\pm 0.043$ )	<b>0.790</b> ( $\pm 0.037$ )
children playing	<b>0.751</b> ( $\pm 0.027$ )	0.685 ( $\pm 0.031$ )	0.736 ( $\pm 0.023$ )
dog bark	<b>0.743</b> ( $\pm 0.040$ )	0.720 ( $\pm 0.033$ )	0.728 ( $\pm 0.040$ )
drilling	<b>0.827</b> ( $\pm 0.027$ )	0.826 ( $\pm 0.026$ )	0.817 ( $\pm 0.026$ )
engine idling	0.767 ( $\pm 0.041$ )	0.733 ( $\pm 0.051$ )	<b>0.791</b> ( $\pm 0.042$ )
jackhammer	0.645 ( $\pm 0.087$ )	0.585 ( $\pm 0.095$ )	<b>0.669</b> ( $\pm 0.059$ )
siren	0.803 ( $\pm 0.062$ )	0.878 ( $\pm 0.034$ )	<b>0.897</b> ( $\pm 0.034$ )
street music	0.792 ( $\pm 0.030$ )	0.782 ( $\pm 0.025$ )	<b>0.812</b> ( $\pm 0.029$ )

**Table 1:** Classification results for the class *gun shot* against the 9 remaining classes.

<sup>2</sup>Urbansound8k dataset

# Geodesic path for Gaussian processes

Gaussian process  $\leftrightarrow$  Kernel  $\leftrightarrow$  PSD.



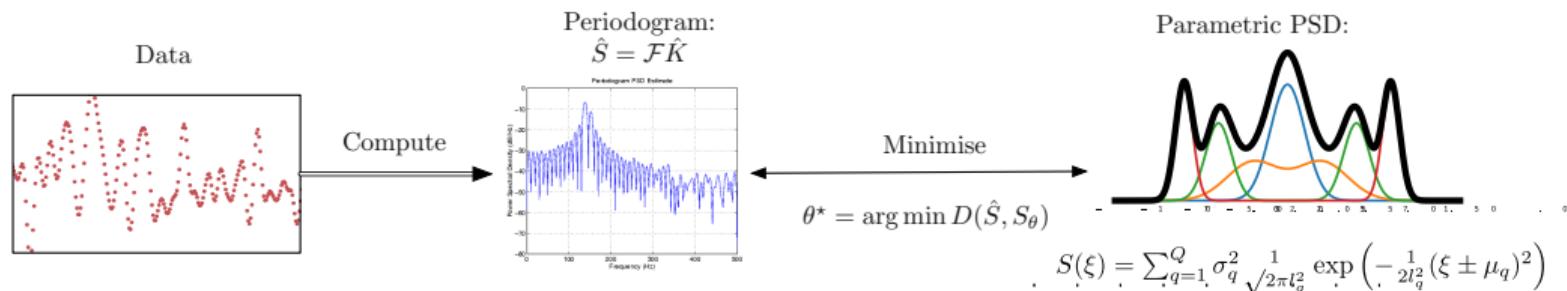
**Spoiler:** GPs can be trained in this way at a linear cost

# How Gaussian processes are trained

**Maximum likelihood:** Standard (very expensive) approach.

**Covariance-based metrics:** Compute sample covariance and apply, e.g.,  $L_p$  distances.

**Frequency-based metrics:** Compute **Periodogram** and use any density-based metric: KL, Bergann, Itakura-Saito, and Wasserstein.



# An interesting case

Let us consider:

- Metric: The *Wasserstein* distance applied to the PSD, i.e.,  $W_2$  on  $S = \mathcal{F}\{K\}$ .
- A Location-scatter family of PSD:  $\left\{ S_{\mu,\sigma}(\xi) = \frac{1}{\sigma} S_{0,1} \left( \frac{\xi - \mu}{\sigma} \right), \mu \in \mathbb{R}, \sigma \in \mathbb{R}_+ \right\}$

## Theorem

For a location-scale family with prototype  $S_{0,1}$ , the minimiser of  $W_2(S, S_{\mu,\sigma})$  is unique, given by

$$\mu^* = \int_0^1 Q(p) dp \quad \text{and} \quad \sigma^* = \frac{1}{\int_0^1 Q_{0,1}^2(p) dp} \int_0^1 Q(p) Q_{0,1}(p) dp \quad (3)$$

where  $Q$  is the quantile function of  $S$ . The PSD  $S$  does not need to be location-scale.

**Corollary:** Training a GP with the Wasserstein distance has a cost  $\mathcal{O}(n)$ . I.e., no need of a gradient flow, as solution is exact and closed form.

# Theoretical aspects

**Does it converge?** I.e., is it true that

$$\theta_n^* = \arg \min D(\hat{S}_n, S_\theta) \xrightarrow[n \rightarrow \infty]{a.s.} \theta^* = \arg \min D(S, S_\theta) \quad (4)$$

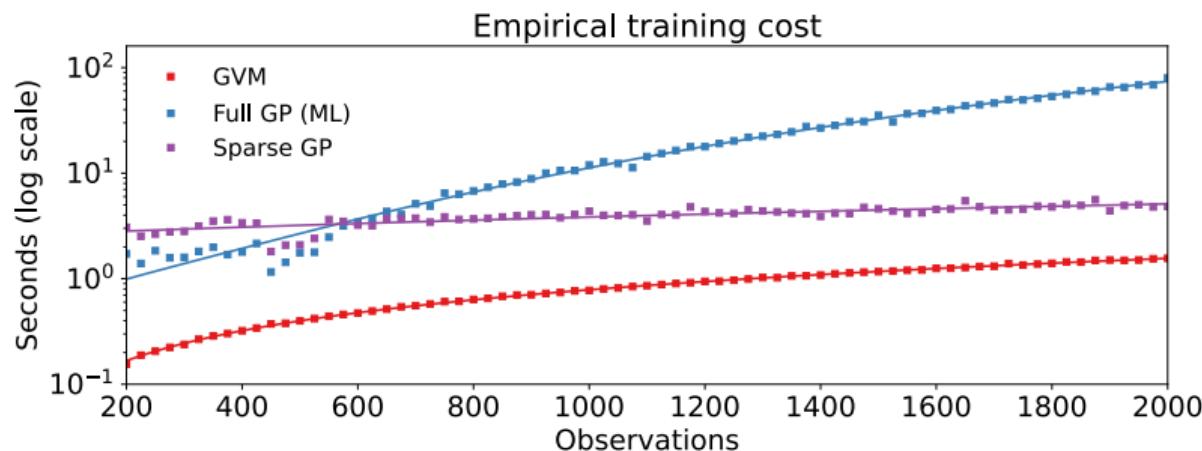
yes it is, provided that:

- **Metric.**  $D$  is either the Wasserstein- $p$  or the  $L_p$  distances with  $p \in \{1, 2\}$
- **Estimator of PSD.**  $D(\hat{S}_n, S) \xrightarrow[n \rightarrow \infty]{a.s.} 0$
- **Identifiability.**  $\theta_n \xrightarrow[n \rightarrow \infty]{} \theta \iff D(S_{\theta_n}, S_\theta) \rightarrow 0;$
- **Compactness.** the parameter space  $\Theta$  is compact.

\*\* This applies to temporal (covariance) distances too

## OT-powered GP training: Linear complexity

- Computation time vs number of observations
- Exact case ( $W_2$  distance and location-scale family)
- **Unevenly-sampled** observations from a single component SM kernel ( $\mu = 0.05, \sigma = 0.01$ ) in the range [0, 1000]
- Compared against: ML estimate starting from the OT value (full GP, 100 iterations), and sparse GP using 200 pseudo inputs



## What we did not see

- Dual formulation of the Kantorovich problem
- Computational OT
- Multimarginal OT
- Unbalanced OT
- Partial OT
- Weak OT
- Particular cases with closed form

# Conclusions & the future

- OT is now in the toolkit for many fields spanning **data analysis, machine learning**, data science and AI.
- OT defines a meaningful distance between distributions, and gives a procedure for **moving particles to minimise such distance**
- Some open challenges:
  - computational **complexity**
  - **curse of dimensionality:** samples for approximations grow exponentially with the dimension
  - **robustness of the solution** with statistical guarantees (noise? outliers?)
  - OT on **different spaces** than Euclidean ones
  - adding some extra constraints (such as temporal consistency)

# A hands-on tutorial on Optimal Transport

Felipe Tobar

Department of Mathematics & I-X  
Imperial College London

22 May, 2025

## Appendix: what we didn't see

# Dual formulation

Recall the primal formulation:  $\text{OT}(\mu, \nu) = \inf_{\pi \in \Pi(\mu, \nu)} \iint c(x, y) d\pi(x, y)$

## Dual problem

$$\text{OT}(\mu, \nu) = \sup_{(\phi, \psi) \in \Phi_c} \left( \int_{\mathcal{X}} \phi d\mu + \int_{\mathcal{X}} \psi d\nu \right),$$

where

$$\Phi_c := \{(\phi, \psi) \in L_1(\mu) \times L_1(\nu), \text{ s.t. } \phi(x) + \psi(y) \leq c(x, y)\}.$$

- $\phi$  and  $\psi$  are scalar function also known as **Kantorovich potentials**
- Primal-dual relationship: the support of  $\pi \in \Pi^*(\mu, \nu)$  is such that  $\phi(x) + \psi(y) = c(x, y)$ .

In the discrete setting:

$$\int_{\mathcal{X}} \phi d \left( \sum_{i=1}^n \mu_i \delta_{x_i} \right) + \int_{\mathcal{X}} \psi d \left( \sum_{j=1}^m \nu_j \delta_{y_j} \right) = \sum_{i=1}^n \mu_i \underbrace{\phi(x_i)}_{\alpha_i} + \sum_{j=1}^m \nu_j \underbrace{\psi(y_j)}_{\beta_j}$$

and  $\Phi_c$  becomes  $\{(\alpha, \beta) \in \mathbb{R}^n \times \mathbb{R}^m \text{ s.t. } \alpha_i + \beta_j \leq c(x_i, y_j)\}$

# Interpretation of Kantorovich duality (discrete)

$$\text{OT}(\mu, \nu) = \min_{\pi \in \Pi(\mu, \nu)} \langle C, \pi \rangle = \max_{(\alpha, \beta) \in D_c} \langle \alpha, \mu \rangle + \langle \beta, \nu \rangle$$

with

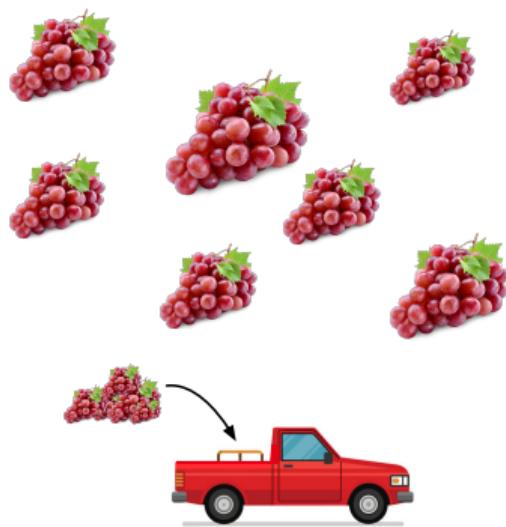
$$D_c := \{(\alpha, \beta) \in \mathbb{R}^n \times \mathbb{R}^m \text{ such that } \forall (i, j) \in \{1, \dots, n\} \times \{1, \dots, m\}, \alpha_i + \beta_j \leq C_{ij}\}$$



# Intuition: the shipper's problem

One vendor sets the following:

- $\alpha_i$  = price for **loading** a kilo of grapes at place  $x_i$  (no matter which plan it goes)
- $\beta_j$  = price for **unloading** a kilo of grapes at place  $y_j$  (no matter from which vineyard it came from)



## Intuition: the shipper's problem

- There are exactly  $\mu_i$  units at vineyard  $x_i$  and  $\nu_j$  needed at plant  $y_j$ ; the vendor asks the price (that she wants to maximize!)

$$\langle \alpha, \mu \rangle + \langle \beta, \nu \rangle$$

- Negative price are allowed!
- Does the vendor have a competitive offer? Her pricing scheme implies that transferring one kilo of grapes from vineyard  $x_i$  to plant  $y_j$  costs exactly  $\alpha_i + \beta_j$ .

.

# Intuition: the shipper's problem

- There are exactly  $\mu_i$  units at vineyard  $x_i$  and  $\nu_j$  needed at plant  $y_j$ ; the vendor asks the price (that she wants to maximize!)

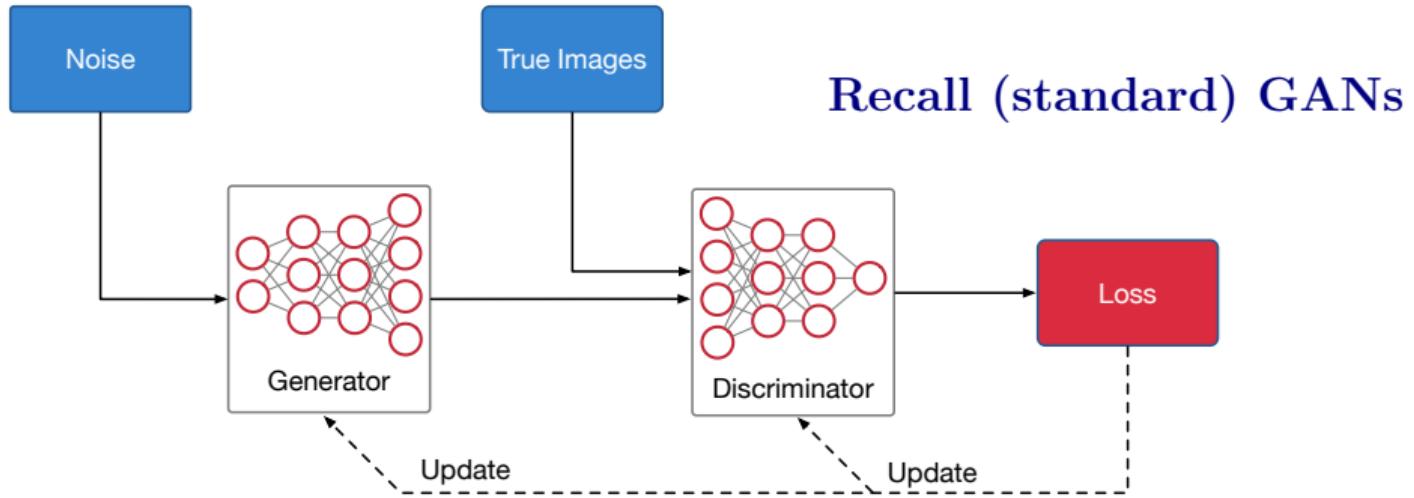
$$\langle \alpha, \mu \rangle + \langle \beta, \nu \rangle$$

- Negative price are allowed!
- Does the vendor have a competitive offer? Her pricing scheme implies that transferring one kilo of grapes from vineyard  $x_i$  to plant  $y_j$  costs exactly  $\alpha_i + \beta_j$ .
- Recall the primal problem: the cost of shipping one unit from  $x_i$  to  $y_j$  is  $C_{i,j}$ .
- Feasible deal for the vendor requires that  $\alpha_i + \beta_j \leq C_{i,j}$ .
- The winery checks that the vendor proposition is a better deal by

$$\sum_{i,j} \pi_{ij} C_{ij} \geq \sum_{i,j} \pi_{ij} (\alpha_j + \beta_j) = \left( \sum_i \alpha_i \sum_j \pi_{ij} \right) + \left( \sum_j \beta_j \sum_i \pi_{ij} \right) = \langle \alpha, \mu \rangle + \langle \beta, \nu \rangle$$

Critically, when  $c(x, y) = |x - y|$ ,  $\alpha = -\beta$ , therefore  $\text{OT}(\mu, \nu) = \max_{\alpha} \langle \alpha, \mu \rangle - \langle \alpha, \nu \rangle$

## Example 5: Wasserstein GANs



$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{\text{data}}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))]$$

Notice the remarkable similarity between the objectives of the (dual) OT formulation and GANs

# Example 5: Wasserstein GANs

GANs vs WGANs: Implementation details

- Discriminator loss no longer a likelihood fn
- Optimised with RMSProp
- Loss for  $D$  and  $G$  have the same form (Kantorovich potential,  $p = 1$ )
- Discriminator's inner loop training  $n_{\text{critic}}$  no longer equal to 1
- Learned parameters are clipped to ensure  $\|f\|_L = 1$



GAN



WGAN

Notebooks: [gan.ipynb](#) & [wgan.ipynb](#)