Advances in Optimal Transport-based Machine Learning

Hongteng Xu

Gaoling School of Artificial Intelligence, Renmin University of China
Beijing Key Laboratory of Big Data Management and Analysis Methods

August 19, 2023

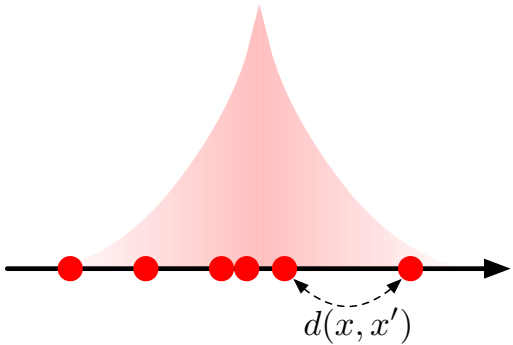
中國人民大學
RENMIN UNIVERSITY OF CHINA

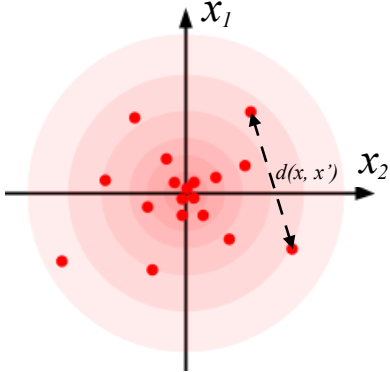
高瓴人工智能学院
Gaoling School of Artificial Intelligence

Outline

Outline

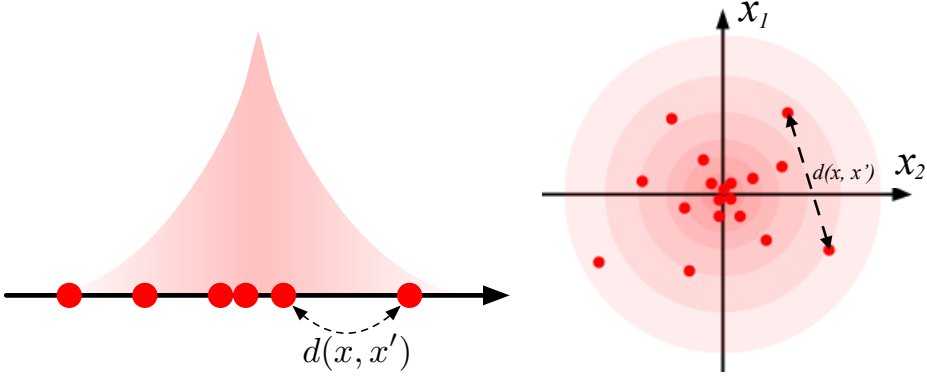
A General Scenario of ML: Metric-Measure Space





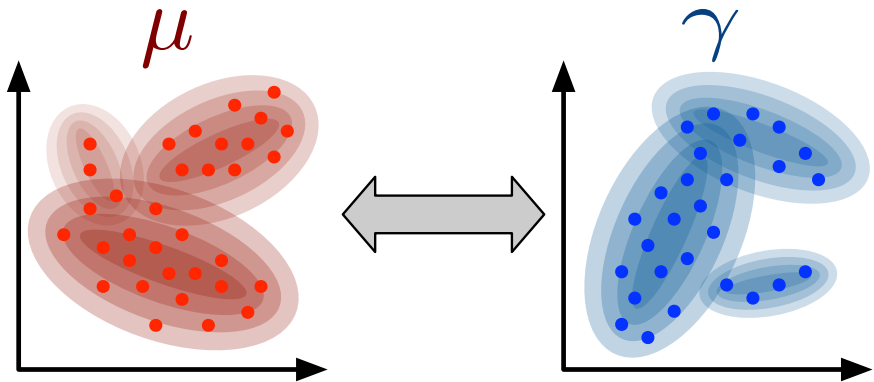
- $\mathcal{X}_{d,\mu} := (\mathcal{X}, d, \mu)$: A metric-measure space, where $x \in \mathcal{X}$ is a sample in the space.
- d: A distance metric of samples (e.g., Euclidean distance).
- $\mathbb{P}$: A space of (probability) measures defined on $\mathcal{X}$.
- $\mu \in \mathbb{P}$: a probability measure on $\mathcal{X}$.

A General Scenario of ML: Metric-Measure Space



- $\mathcal{X}_{d,\mu} := (\mathcal{X}, d, \mu)$: A metric-measure space, where $x \in \mathcal{X}$ is a sample in the space.
- Most ML tasks are constructing/reconstructing mm-spaces from observed data:
  - Data representation: Find a map $f : \mathcal{X}_{d,\mu} \mapsto \mathcal{Z}_{d',\mu'}$.
  - Metric learning: Learn a (pseudo-)metric d.
  - Generative modeling: Estimate μ by a model g for $\mathcal{X}$.

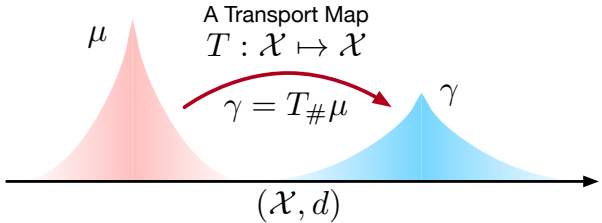
Distribution Comparison: The Key Machine Learning Task



- Data Clustering, Domain Adaptation, Generative Modeling, Evaluation of Generative Model, ...

Origin: The Monge-form of The Optimal Transport Problem

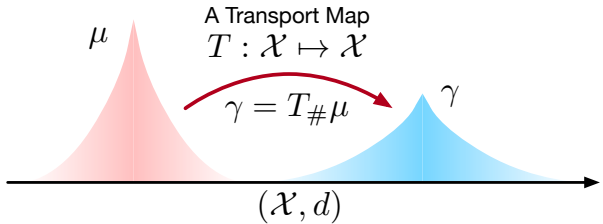
Gaspard Monge (1746-1818)



The Monge-form of OT problem proposed in 1781.

Origin: The Monge-form of The Optimal Transport Problem

Gaspard Monge (1746-1818)



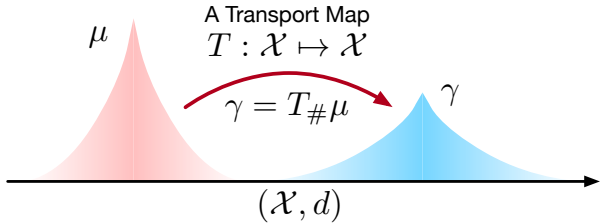
The Monge-form of OT problem proposed in 1781.

- $T_{\#}\mu$: The **push-forward** of μ, for $\mathcal{S} \subset \mathcal{X}$

$$T_{\#}\mu(\mathcal{S}) = \mu(\{x : T(x) \in \mathcal{S}\}) = \mu(T^{-1}(\mathcal{S})). \tag{1}$$

Origin: The Monge-form of The Optimal Transport Problem

Gaspard Monge (1746-1818)



The Monge-form of OT problem proposed in 1781.

- $T_{\#}\mu$: The **push-forward** of μ, for $\mathcal{S} \subset \mathcal{X}$

$$T_{\#}\mu(\mathcal{S}) = \mu(\{x : T(x) \in \mathcal{S}\}) = \mu(T^{-1}(\mathcal{S})). \tag{1}$$

- Find a measure-preserving map T to minimize the cost of moving samples:

$$M_p(\mu, \gamma) := \Big(\underbrace{\inf_{T : T_{\#}\mu = \gamma}}_{\text{measure preserving}} \int_{x \in \mathcal{X}} \underbrace{d^p(x, T(x))}_{\text{cost per sample}} \, \mathrm{d}\mu(x) \Big)^{1/p}. \tag{2}$$

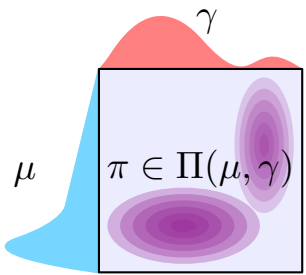
From Transport Map to Transport Plan: The Kantorovich-form of OT

Note that, the minimizer of (2) may not exist!

From Transport Map to Transport Plan: The Kantorovich-form of OT

Note that, the minimizer of (2) may not exist!

Leonid Kantorovich (1912-1986)

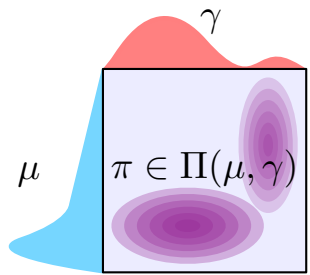


The Kantorovich-form of OT proposed in 1939

From Transport Map to Transport Plan: The Kantorovich-form of OT

Note that, the minimizer of (2) may not exist!

Leonid Kantorovich (1912-1986)



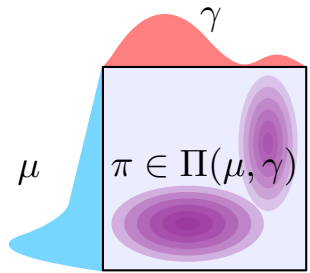
The Kantorovich-form of OT proposed in 1939

- $\Pi(\mu, \gamma) = \{\pi > 0 | \int_x \pi(x, y)dx = \gamma(y), \int_y \pi(x, y)dy = \mu(x)\}$ include all joint distributions taking μ and γ as marginals.
- $\pi \in \Pi(\mu, \gamma)$ is called **transport plan or coupling**.

From Transport Map to Transport Plan: The Kantorovich-form of OT

Note that, the minimizer of (2) may not exist!

Leonid Kantorovich (1912-1986)



The Kantorovich-form of OT proposed in 1939

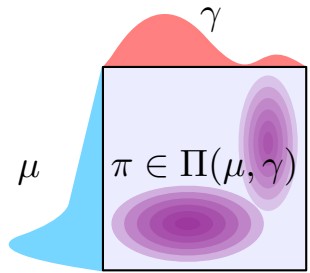
- $\Pi(\mu, \gamma) = \{\pi > 0 | \int_x \pi(x, y)dx = \gamma(y), \int_y \pi(x, y)dy = \mu(x)\}$ include all joint distributions taking μ and γ as marginals.
- $\pi \in \Pi(\mu, \gamma)$ is called **transport plan or coupling**.
- Find an optimal transport plan to minimize the expected cost.

$$W_p(\mu, \gamma) := \Big(\inf_{\pi \in \Pi(\mu,\gamma)} \int_{(x,y)\in\mathcal{X}^2} d^p(x, y) \mathrm{d}\pi(x, y) \Big)^{1/p}$$

From Transport Map to Transport Plan: The Kantorovich-form of OT

Note that, the minimizer of (2) may not exist!

Leonid Kantorovich (1912-1986)



The Kantorovich-form of OT proposed in 1939

- $\Pi(\mu, \gamma) = \{\pi > 0 | \int_x \pi(x, y)dx = \gamma(y), \int_y \pi(x, y)dy = \mu(x)\}$ include all joint distributions taking μ and γ as marginals.
- $\pi \in \Pi(\mu, \gamma)$ is called **transport plan or coupling**.
- Find an optimal transport plan to minimize the expected cost.

$$W_p(\mu, \gamma) := \Big(\inf_{\pi \in \Pi(\mu,\gamma)} \int_{(x,y)\in\mathcal{X}^2} d^p(x, y)\mathrm{d}\pi(x, y)\Big)^{1/p} = \inf_{\pi \in \Pi(\mu,\gamma)} \mathbb{E}^{1/p}_{x,y\sim\pi}[d^p(x, y)]. \quad (3)$$

From Transport Map to Transport Plan: The Kantorovich-form of OT

Relations to the Monge-form OT:

- Applying the transport plan π, we allow each sample $x \sim \mu$ to be split and mapped to multiple locations.
- If the optimal transport map T^* exists, it determines a transport plan π, so

$$W_p(\mu, \gamma) \leq M_p(\mu, \gamma). \tag{4}$$

From Transport Map to Transport Plan: The Kantorovich-form of OT

Relations to the Monge-form OT:

- Applying the transport plan π, we allow each sample $x \sim \mu$ to be split and mapped to multiple locations.
- If the optimal transport map T^* exists, it determines a transport plan π, so

$$W_p(\mu, \gamma) \leq M_p(\mu, \gamma). \tag{4}$$

When $d(x, y) = \|x - y\|_p$, W_p is called p-**Wasserstein distance**.

From Transport Map to Transport Plan: The Kantorovich-form of OT

Relations to the Monge-form OT:

- Applying the transport plan π, we allow each sample $x \sim \mu$ to be split and mapped to multiple locations.
- If the optimal transport map T^* exists, it determines a transport plan π, so

$$W_p(\mu, \gamma) \leq M_p(\mu, \gamma). \tag{4}$$

When $d(x, y) = \|x - y\|_p$, W_p is called p-**Wasserstein distance**.

- When $p = 1$, $d(x, y) = |x - y|$, W_1 is the **Earth Mover Distance (EMD)**.

From Transport Map to Transport Plan: The Kantorovich-form of OT

Relations to the Monge-form OT:

- Applying the transport plan π, we allow each sample $x \sim \mu$ to be split and mapped to multiple locations.
- If the optimal transport map T^* exists, it determines a transport plan π, so

$$W_p(\mu, \gamma) \leq M_p(\mu, \gamma). \tag{4}$$

When $d(x, y) = \|x - y\|_p$, W_p is called p-**Wasserstein distance**.

- When $p = 1$, $d(x, y) = |x - y|$, W_1 is the **Earth Mover Distance (EMD)**.
- When $p = 2$, $\mu = \mathcal{N}(x_1, \Sigma_1)$ and $\gamma = \mathcal{N}(x_2, \Sigma_2)$,

$$W_2(\mu, \gamma) = (\|x_1 - x_2\|_2^2 + \mathsf{tr}(\Sigma_1) + \mathsf{tr}(\Sigma_2) - 2\mathsf{tr}((\Sigma_1^{1/2}\Sigma_2\Sigma_1^{1/2})^{1/2}))^{1/2}. \tag{5}$$

From Transport Map to Transport Plan: The Kantorovich-form of OT

Relations to the Monge-form OT:

- Applying the transport plan π, we allow each sample $x \sim \mu$ to be split and mapped to multiple locations.
- If the optimal transport map T^* exists, it determines a transport plan π, so

$$W_p(\mu, \gamma) \leq M_p(\mu, \gamma). \tag{4}$$

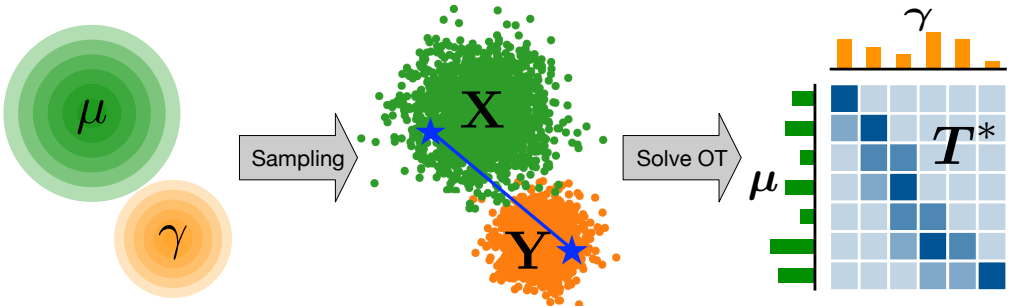
When $d(x, y) = \|x - y\|_p$, W_p is called p-**Wasserstein distance**.

- When $p = 1$, $d(x, y) = |x - y|$, W_1 is the **Earth Mover Distance (EMD)**.
- When $p = 2$, $\mu = \mathcal{N}(x_1, \Sigma_1)$ and $\gamma = \mathcal{N}(x_2, \Sigma_2)$,

$$W_2(\mu, \gamma) = (\|x_1 - x_2\|_2^2 + \mathrm{tr}(\Sigma_1) + \mathrm{tr}(\Sigma_2) - 2\mathrm{tr}((\Sigma_1^{1/2}\Sigma_2\Sigma_1^{1/2})^{1/2}))^{1/2}. \tag{5}$$

W_p is a valid metric for probability measures.

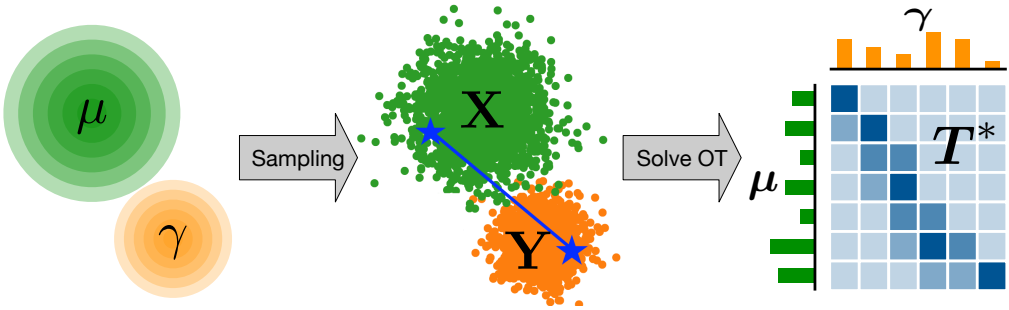
Empirical OT Problem Defined on Samples



Given $\boldsymbol{X} = \{x_m\}_{m=1}^M \sim \mu$, $\boldsymbol{Y} = \{y_n\}_{n=1}^N \sim \gamma$, $\boldsymbol{\mu} \in \Delta^{N-1}$ and $\boldsymbol{\gamma} \in \Delta^{M-1}$,

$$\widehat{W}_p(\boldsymbol{X}, \boldsymbol{Y}) := \Big(\min_{\boldsymbol{T} \in \Pi(\boldsymbol{\mu}, \boldsymbol{\gamma})} \sum_{m=1}^M \sum_{n=1}^N d^p(x_m, y_n) t_{mn} \Big)^{1/p}$$

Empirical OT Problem Defined on Samples

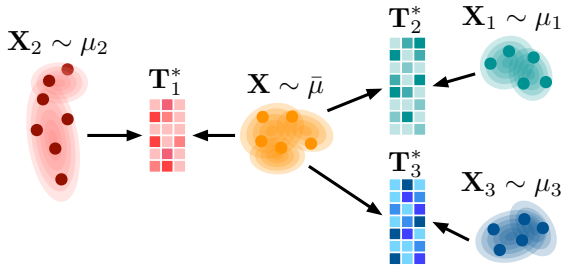


Given $\boldsymbol{X} = \{x_m\}_{m=1}^M \sim \mu$, $\boldsymbol{Y} = \{y_n\}_{n=1}^N \sim \gamma$, $\boldsymbol{\mu} \in \Delta^{N-1}$ and $\boldsymbol{\gamma} \in \Delta^{M-1}$,

$$\begin{aligned}\widehat{W}_p(\boldsymbol{X}, \boldsymbol{Y}) :=& \Big(\min_{\boldsymbol{T} \in \Pi(\boldsymbol{\mu}, \boldsymbol{\gamma})} \sum\nolimits_{m=1}^{M} \sum\nolimits_{n=1}^{N} d^p(x_m, y_n) t_{mn}\Big)^{1/p} \\ =& \Big(\min_{\boldsymbol{T} \in \Pi(\boldsymbol{\mu}, \boldsymbol{\gamma})} \langle \boldsymbol{D}, \boldsymbol{T} \rangle\Big)^{1/p} = \min_{\boldsymbol{T} \in \Pi(\boldsymbol{\mu}, \boldsymbol{\gamma})} \mathbb{E}^{1/p}_{(x,y) \sim \boldsymbol{T}}[d^p(x, y)],\end{aligned} \tag{6}$$

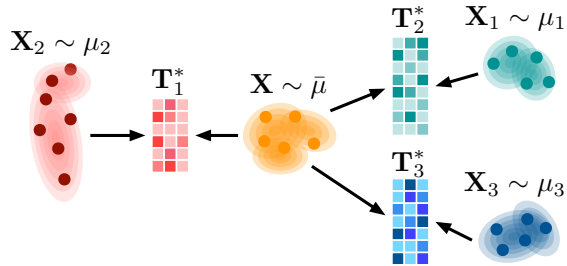
where $\boldsymbol{D} = [d^p(x_m, y_n)]$, $\boldsymbol{T} = [t_{mn}]$, $\Pi(\boldsymbol{\mu}, \boldsymbol{\gamma}) = \{\boldsymbol{T} > \mathbf{0} | \boldsymbol{T}\mathbf{1}_M = \boldsymbol{\mu}, \boldsymbol{T}^\top \mathbf{1}_N = \boldsymbol{\gamma}\}$.

Wasserstein Barycenters



- Denote $\mathcal{P}_{\mathcal{X}_d}$ as the space of all probability measures in the metric space $\mathcal{X}_d$.
- $(\mathcal{P}_{\mathcal{X}_d}, W_p)$ becomes a metric space of probability measures.

Wasserstein Barycenters



- Denote $\mathcal{P}_{\mathcal{X}_d}$ as the space of all probability measures in the metric space $\mathcal{X}_d$.
- $(\mathcal{P}_{\mathcal{X}_d}, W_p)$ becomes a metric space of probability measures.
- Given a set of probability measures $\{\mu_k\}_{k=1}^K \subset \mathcal{P}_{\mathcal{X}_d}$, we can define the p-**Wasserstein barycenter** [Agueh et al, 2021] as

$$\bar{\mu} := \arg\min_{\mu \in \mathcal{P}_{\mathcal{X}_d}} \sum_{k=1}^{K} W_p^p(\mu, \mu_k). \tag{7}$$

[Agueh et al, 2021] Agueh, M. and Carlier, G., Barycenters in the Wasserstein space. SIAM Journal on Mathematical Analysis, 2011.

Advantages of Optimal Transport

A valid metric for probability measures

- Apply to distribution comparison and fitting
- Wasserstein barycenter can achieve multi-distribution averaging and fusion

Advantages of Optimal Transport

A valid metric for probability measures

- Apply to distribution comparison and fitting
- Wasserstein barycenter can achieve multi-distribution averaging and fusion

Consistent sample-based estimation

- With the increase of samples, $\widehat{W}_p \to W_p$

Advantages of Optimal Transport

A valid metric for probability measures

- Apply to distribution comparison and fitting
- Wasserstein barycenter can achieve multi-distribution averaging and fusion

Consistent sample-based estimation

- With the increase of samples, $\widehat{W}_p \to W_p$

The OT plan/matrix indicates the coherency of sample pairs

- Apply to point cloud matching and registration

Computational Bottlenecks of Optimal Transport and Possible Solutions

- A constrained linear programming problem:

$$\widehat{W}_p^p(\boldsymbol{X}, \boldsymbol{Y}) = \min_{\boldsymbol{T} \in \Pi(\boldsymbol{\mu}, \boldsymbol{\gamma})} \langle \boldsymbol{D}, \boldsymbol{T} \rangle. \tag{8}$$

Apply the Simplex algorithm leads to $\mathcal{O}(N^3)$ complexity.

Computational Bottlenecks of Optimal Transport and Possible Solutions

- A constrained linear programming problem:

$$\widehat{W}_p^p(\boldsymbol{X}, \boldsymbol{Y}) = \min_{\boldsymbol{T} \in \Pi(\boldsymbol{\mu}, \boldsymbol{\gamma})} \langle \boldsymbol{D}, \boldsymbol{T} \rangle. \tag{8}$$

  Apply the Simplex algorithm leads to $\mathcal{O}(N^3)$ complexity.

- Solution 1: Develop efficient optimization algorithms and acceleration methods
  - Sinkhorn-scaling
  - Proximal point
  - Bregman ADMM

Computational Bottlenecks of Optimal Transport and Possible Solutions

- A constrained linear programming problem:

$$\widehat{W}_p^p(\boldsymbol{X}, \boldsymbol{Y}) = \min_{\boldsymbol{T} \in \Pi(\boldsymbol{\mu}, \boldsymbol{\gamma})} \langle \boldsymbol{D}, \boldsymbol{T} \rangle. \tag{8}$$

  Apply the Simplex algorithm leads to $\mathcal{O}(N^3)$ complexity.

- Solution 1: Develop efficient optimization algorithms and acceleration methods
  - Sinkhorn-scaling
  - Proximal point
  - Bregman ADMM
- Solution 2: Apply structured/stochastic OT plan
  - Stochastic optimization
  - Sinkhorn-scaling with importance sparsification

Computational Bottlenecks of Optimal Transport and Possible Solutions

- A constrained linear programming problem:

$$\widehat{W}_p^p(\boldsymbol{X}, \boldsymbol{Y}) = \min_{\boldsymbol{T} \in \Pi(\boldsymbol{\mu}, \boldsymbol{\gamma})} \langle \boldsymbol{D}, \boldsymbol{T} \rangle. \tag{8}$$

  Apply the Simplex algorithm leads to $\mathcal{O}(N^3)$ complexity.

- Solution 1: Develop efficient optimization algorithms and acceleration methods
  - Sinkhorn-scaling
  - Proximal point
  - Bregman ADMM
- Solution 2: Apply structured/stochastic OT plan
  - Stochastic optimization
  - Sinkhorn-scaling with importance sparsification
- Solution 3: Explore efficient surrogates of OT distance
  - Sliced Wasserstein (SW) distance
  - Hilbert curve projection (HCP) distance

Sinkhorn-scaling Algorithm for Entropic OT

Motivation and Principle:

- Improve the smoothness of the OT problem

Sinkhorn-scaling Algorithm for Entropic OT

Motivation and Principle:

- Improve the smoothness of the OT problem

Sinkhorn Distance (Entropic OT Problem) [Cuturi, NeurIPS 2013]

$$\widehat{W}_{p,\epsilon} := \min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu},\boldsymbol{\gamma})} \langle \boldsymbol{D}, \boldsymbol{T}\rangle - \epsilon \underbrace{\mathsf{H}(\boldsymbol{T})}_{\text{Entropy}},$$

$$\mathsf{H}(\boldsymbol{T}) = -\langle \log \boldsymbol{T} - \mathbf{1}, \boldsymbol{T}\rangle.$$

Sinkhorn-scaling Algorithm for Entropic OT

Motivation and Principle:

- Improve the smoothness of the OT problem

Sinkhorn Distance (Entropic OT Problem) [Cuturi, NeurIPS 2013]

$$\widehat{W}_{p,\epsilon} := \min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu},\boldsymbol{\gamma})} \langle \boldsymbol{D}, \boldsymbol{T}\rangle - \epsilon \underbrace{\mathsf{H}(\boldsymbol{T})}_{\text{Entropy}}, \tag{9}$$

$$\mathsf{H}(\boldsymbol{T}) = -\langle \log \boldsymbol{T} - \mathbf{1}, \boldsymbol{T}\rangle.$$

Sinkhorn-Knopp algorithm:

1. Set a kernel matrix $\boldsymbol{\Phi} = \exp(-\frac{\boldsymbol{D}}{\epsilon})$ and a dual variable $\boldsymbol{a} = \boldsymbol{\mu}$.
2. **Sinkhorn iteration:** Repeat $\boldsymbol{b} = \frac{\boldsymbol{\gamma}}{\boldsymbol{\Phi}^\top \boldsymbol{a}}$ and $\boldsymbol{a} = \frac{\boldsymbol{\mu}}{\boldsymbol{\Phi}\boldsymbol{b}}$ until convergence.
3. $\boldsymbol{T}^* = \boldsymbol{\Phi} \odot (\boldsymbol{a}\boldsymbol{b}^\top)$.

[Cuturi, NeurIPS 2013] Cuturi, M. Sinkhorn distances: Lightspeed computation of optimal transport. NeurIPS, 2013.

Proximal Point Algorithm: A Variant of Sinkhorn-scaling

Motivation and Principle:

- The (explicit) entropic regularizer might be unnecessary
- Solve the "exact" OT problem via a Sinkhorn-like algorithm

Proximal Point Algorithm: A Variant of Sinkhorn-scaling

Motivation and Principle:

- The (explicit) entropic regularizer might be unnecessary
- Solve the "exact" OT problem via a Sinkhorn-like algorithm

Proximal point algorithm:

1. Initialize $\boldsymbol{T}^{(0)} = \boldsymbol{\mu}\boldsymbol{\nu}^\top$.

Proximal Point Algorithm: A Variant of Sinkhorn-scaling

Motivation and Principle:

- The (explicit) entropic regularizer might be unnecessary
- Solve the "exact" OT problem via a Sinkhorn-like algorithm

Proximal point algorithm:

1. Initialize $\boldsymbol{T}^{(0)} = \boldsymbol{\mu}\boldsymbol{\nu}^\top$.
2. In the m-th iteration, consider the penalty between the optimal transport and its previous approximation [Xie, et al., UAI 2020]

$$\min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu},\boldsymbol{\gamma})} \langle \boldsymbol{D}, \boldsymbol{T}\rangle + \epsilon \underbrace{\mathsf{KL}(\boldsymbol{T}\|\boldsymbol{T}^{(m)})}_{\text{Proximal term}}$$

Proximal Point Algorithm: A Variant of Sinkhorn-scaling

Motivation and Principle:

- The (explicit) entropic regularizer might be unnecessary
- Solve the "exact" OT problem via a Sinkhorn-like algorithm

Proximal point algorithm:

1. Initialize $\boldsymbol{T}^{(0)} = \boldsymbol{\mu}\boldsymbol{\nu}^\top$.
2. In the m-th iteration, consider the penalty between the optimal transport and its previous approximation [Xie, et al., UAI 2020]

$$\begin{aligned} &\min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu},\boldsymbol{\gamma})} \langle \boldsymbol{D}, \boldsymbol{T}\rangle + \epsilon \underbrace{\mathsf{KL}(\boldsymbol{T}\|\boldsymbol{T}^{(m)})}_{\text{Proximal term}} \\ \Rightarrow &\min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu},\boldsymbol{\gamma})} \langle \underbrace{\boldsymbol{D} - \epsilon \log \boldsymbol{T}^{(m)}}_{\epsilon \log \boldsymbol{\Phi}^{(m)}}, \boldsymbol{T}\rangle - \epsilon \mathsf{H}(\boldsymbol{T}). \end{aligned} \tag{10}$$

Proximal Point Algorithm: A Variant of Sinkhorn-scaling

Motivation and Principle:

- The (explicit) entropic regularizer might be unnecessary
- Solve the "exact" OT problem via a Sinkhorn-like algorithm

Proximal point algorithm:

1. Initialize $\boldsymbol{T}^{(0)} = \boldsymbol{\mu}\boldsymbol{\nu}^\top$.
2. In the m-th iteration, consider the penalty between the optimal transport and its previous approximation [Xie, et al., UAI 2020]

$$\begin{aligned} &\min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu},\boldsymbol{\gamma})} \langle \boldsymbol{D}, \boldsymbol{T}\rangle + \epsilon \underbrace{\mathsf{KL}(\boldsymbol{T}\|\boldsymbol{T}^{(m)})}_{\text{Proximal term}} \\ \Rightarrow &\min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu},\boldsymbol{\gamma})} \langle \underbrace{\boldsymbol{D} - \epsilon \log \boldsymbol{T}^{(m)}}_{\epsilon \log \boldsymbol{\Phi}^{(m)}}, \boldsymbol{T}\rangle - \epsilon \mathsf{H}(\boldsymbol{T}). \end{aligned} \tag{10}$$

3. Apply the Sinkhorn iterations to obtain $\boldsymbol{T}^{(m+1)} = \boldsymbol{\Phi}^{(m)} \odot (\boldsymbol{a}^{(m)}(\boldsymbol{b}^{(m)})^\top)$.

[Xie, et al., UAI 2020] Xie, Y., Wang, X., Wang, R., & Zha, H. A fast proximal point method for computing exact Wasserstein distance. UAI 2020.

Proximal Point Algorithm: A Variant of Sinkhorn-scaling

The connections between the Sinkhorn-scaling and the proximal point:

- In the m-th iteration, denote $\boldsymbol{a}^{(m)}(\boldsymbol{b}^{(m)})^\top$ as $\boldsymbol{\Delta}^{(m)}$:

$$\boldsymbol{\Phi}^{(m)} = \exp\Big(-\frac{\boldsymbol{D} - \epsilon \log \boldsymbol{T}^{(m-1)}}{\epsilon}\Big) = \exp\Big(-\frac{\boldsymbol{D}}{\epsilon}\Big) \odot \boldsymbol{T}^{(m-1)}$$

Proximal Point Algorithm: A Variant of Sinkhorn-scaling

The connections between the Sinkhorn-scaling and the proximal point:

- In the m-th iteration, denote $\boldsymbol{a}^{(m)}(\boldsymbol{b}^{(m)})^\top$ as $\boldsymbol{\Delta}^{(m)}$:

$$\begin{aligned}\boldsymbol{\Phi}^{(m)} &= \exp\Big(-\frac{\boldsymbol{D}-\epsilon\log\boldsymbol{T}^{(m-1)}}{\epsilon}\Big) = \exp\Big(-\frac{\boldsymbol{D}}{\epsilon}\Big)\odot\boldsymbol{T}^{(m-1)}\\ &= \exp\Big(-\frac{\boldsymbol{D}}{\epsilon}\Big)\odot\boldsymbol{\Phi}^{(m-1)}\odot\boldsymbol{\Delta}^{(m-1)}\\ &= \exp\Big(-\frac{\boldsymbol{D}}{\epsilon}\Big)\odot\exp\Big(-\frac{\boldsymbol{D}-\gamma\log\boldsymbol{T}^{(m-2)}}{\epsilon}\Big)\odot\boldsymbol{\Delta}^{(m-1)}\end{aligned}$$

Proximal Point Algorithm: A Variant of Sinkhorn-scaling

The connections between the Sinkhorn-scaling and the proximal point:

- In the m-th iteration, denote $\boldsymbol{a}^{(m)}(\boldsymbol{b}^{(m)})^\top$ as $\boldsymbol{\Delta}^{(m)}$:

$$
\begin{aligned}
\boldsymbol{\Phi}^{(m)} &= \exp\Big(-\frac{\boldsymbol{D}-\epsilon\log\boldsymbol{T}^{(m-1)}}{\epsilon}\Big) = \exp\Big(-\frac{\boldsymbol{D}}{\epsilon}\Big)\odot\boldsymbol{T}^{(m-1)} \\
&= \exp\Big(-\frac{\boldsymbol{D}}{\epsilon}\Big)\odot\boldsymbol{\Phi}^{(m-1)}\odot\boldsymbol{\Delta}^{(m-1)} \\
&= \exp\Big(-\frac{\boldsymbol{D}}{\epsilon}\Big)\odot\exp\Big(-\frac{\boldsymbol{D}-\gamma\log\boldsymbol{T}^{(m-2)}}{\epsilon}\Big)\odot\boldsymbol{\Delta}^{(m-1)} \\
&= \exp\Big(-\frac{m}{\epsilon}\boldsymbol{D}\Big)\odot\underbrace{\big(\odot_{i=0}^{m-1}\boldsymbol{\Delta}^{(i)}\big)}_{\boldsymbol{\Delta}_m}.
\end{aligned}
\tag{11}
$$

- $\boldsymbol{\Delta}_m$ determines the initial point while the problem corresponding to the iteration steps is convex.
- So proximal point algorithm implements the Sinkhorn-scaling with a decaying weight $\frac{\epsilon}{m}$.

Bregman ADMM: Solve OT without Sinkhorn

Motivation and Principle:

- The Sinkhorn-based algorithm often suffers from numerical instability issue.
- The Sinkhorn-based algorithm is restricted to the OT problem with entropy or KLD regularization.

Bregman ADMM: Solve OT without Sinkhorn

Motivation and Principle:

- The Sinkhorn-based algorithm often suffers from numerical instability issue.
- The Sinkhorn-based algorithm is restricted to the OT problem with entropy or KLD regularization.
- Decouple the doubly-stochastic constraint to two one-side constraints can simplify the problem.

Bregman ADMM: Solve OT without Sinkhorn

Motivation and Principle:

- The Sinkhorn-based algorithm often suffers from numerical instability issue.
- The Sinkhorn-based algorithm is restricted to the OT problem with entropy or KLD regularization.
- Decouple the doubly-stochastic constraint to two one-side constraints can simplify the problem.

Reformulation of OT problem [Wang, et al. NeurIPS 2014]:

- Introduce an auxiliary variable $\boldsymbol{S}$ and a dual variable $\boldsymbol{Z}$:

$$\min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu},\boldsymbol{\gamma})}\langle \boldsymbol{D},\boldsymbol{T}\rangle \Leftrightarrow \min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu},\cdot),\boldsymbol{S}\in\Pi(\cdot,\boldsymbol{\gamma}),\boldsymbol{T}=\boldsymbol{S}}\langle \boldsymbol{D},\boldsymbol{T}\rangle$$

Bregman ADMM: Solve OT without Sinkhorn

Motivation and Principle:

- The Sinkhorn-based algorithm often suffers from numerical instability issue.
- The Sinkhorn-based algorithm is restricted to the OT problem with entropy or KLD regularization.
- Decouple the doubly-stochastic constraint to two one-side constraints can simplify the problem.

Reformulation of OT problem [Wang, et al. NeurIPS 2014]:

- Introduce an auxiliary variable $\boldsymbol{S}$ and a dual variable $\boldsymbol{Z}$:

$$\begin{aligned} &\min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu},\boldsymbol{\gamma})}\langle \boldsymbol{D},\boldsymbol{T}\rangle \Leftrightarrow \min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu},\cdot),\boldsymbol{S}\in\Pi(\cdot,\boldsymbol{\gamma}),\boldsymbol{T}=\boldsymbol{S}}\langle \boldsymbol{D},\boldsymbol{T}\rangle \\ &\Leftrightarrow \min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu},\cdot),\boldsymbol{S}\in\Pi(\cdot,\boldsymbol{\gamma}),\boldsymbol{Z}}\langle \boldsymbol{D},\boldsymbol{T}\rangle + \underbrace{\langle \boldsymbol{Z},\boldsymbol{T}-\boldsymbol{S}\rangle + \overbrace{\epsilon \mathsf{B}_\phi(\boldsymbol{T},\boldsymbol{S})}^{\text{Bregman Div.}}}_{\text{Augmented Lagrangian}} \end{aligned} \tag{12}$$

[Wang, et al. NeurIPS 2014] Wang, H., & Banerjee, A. Bregman alternating direction method of multipliers. NeurIPS 2014.

Bregman ADMM: Solve OT without Sinkhorn

Bregman Divergence: Given a differentiable and strictly convex function ϕ,

$$B_\phi(x, y) = \phi(x) - \phi(y) - \langle \nabla\phi(y), x - y \rangle. \tag{13}$$

Bregman ADMM: Solve OT without Sinkhorn

Bregman Divergence: Given a differentiable and strictly convex function ϕ,

$$B_\phi(x, y) = \phi(x) - \phi(y) - \langle \nabla\phi(y), x - y \rangle. \tag{13}$$

Commonly-used Bregman divergence:

- $\phi(x) = \frac{1}{2}x^2$: Euclidean distance $B_\phi(x, y) = \frac{1}{2}\|x - y\|^2$.
- $\phi(x) = x \log x - x$: KL-divergence $B_\phi(x, y) = \mathsf{KL}(x\|y) = x \log \frac{x}{y} - x + y$.

Bregman ADMM: Solve OT without Sinkhorn

Bregman Divergence: Given a differentiable and strictly convex function ϕ,

$$B_\phi(x, y) = \phi(x) - \phi(y) - \langle \nabla\phi(y), x - y \rangle. \tag{13}$$

Commonly-used Bregman divergence:

- $\phi(x) = \frac{1}{2}x^2$: Euclidean distance $B_\phi(x, y) = \frac{1}{2}\|x - y\|^2$.
- $\phi(x) = x \log x - x$: KL-divergence $B_\phi(x, y) = \mathsf{KL}(x\|y) = x \log \frac{x}{y} - x + y$.

Naturally, the Bregman ADMM is also applicable for various regularized OT:

- Considering the above Bregman divergence leads to the OT problems with entropic or quadratic regularizers.

Bregman ADMM: Solve OT without Sinkhorn

Bregman Divergence: Given a differentiable and strictly convex function ϕ,

$$B_\phi(x, y) = \phi(x) - \phi(y) - \langle \nabla\phi(y), x - y \rangle. \tag{13}$$

Commonly-used Bregman divergence:

- $\phi(x) = \frac{1}{2}x^2$: Euclidean distance $B_\phi(x, y) = \frac{1}{2}\|x - y\|^2$.
- $\phi(x) = x \log x - x$: KL-divergence $B_\phi(x, y) = \mathsf{KL}(x\|y) = x \log \frac{x}{y} - x + y$.

Naturally, the Bregman ADMM is also applicable for various regularized OT:

- Considering the above Bregman divergence leads to the OT problems with entropic or quadratic regularizers.

The Bregman ADMM algorithm solves the OT problems iteratively.

- Each step has a closed form.
- Sublinear convergence rate.

Bregman ADMM: Solve OT without Sinkhorn

- Initialize $\boldsymbol{Z}^{(0)} = \mathbf{0}$, $\boldsymbol{T}^{(0)} = \boldsymbol{S}^{(0)} = \boldsymbol{\mu}\boldsymbol{\gamma}^\top$.

Bregman ADMM: Solve OT without Sinkhorn

- Initialize $\boldsymbol{Z}^{(0)} = \mathbf{0}$, $\boldsymbol{T}^{(0)} = \boldsymbol{S}^{(0)} = \boldsymbol{\mu}\boldsymbol{\gamma}^\top$.
- Repeat the following steps till convergence:
  1. Update primal variable $\boldsymbol{T}$:

$$\boldsymbol{T}^{(m+1)} = \arg\min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu},\cdot)} \langle \boldsymbol{D}, \boldsymbol{T}\rangle + \langle \boldsymbol{Z}^{(m)}, \boldsymbol{T}\rangle + \epsilon \mathsf{KL}(\boldsymbol{T}\|\boldsymbol{S}^{(m)})$$

$$\xrightarrow{\text{First-order optimality}} \boldsymbol{T}^{(m+1)} = \exp\Big(\frac{\epsilon \log \boldsymbol{S}^{(m)} - \boldsymbol{D} - \boldsymbol{Z}^{(m)}}{\epsilon}\Big)$$

$$\xrightarrow{\text{Project to } \Pi(\boldsymbol{\mu},\cdot)} \boldsymbol{T}^{(m+1)} = \mathsf{diag}(\boldsymbol{\mu})\mathsf{Softmax}_r\Big(\frac{\epsilon \log \boldsymbol{S}^{(m)} - \boldsymbol{D} - \boldsymbol{Z}^{(m)}}{\epsilon}\Big).$$

Bregman ADMM: Solve OT without Sinkhorn

- Initialize $\boldsymbol{Z}^{(0)} = \mathbf{0}$, $\boldsymbol{T}^{(0)} = \boldsymbol{S}^{(0)} = \boldsymbol{\mu}\boldsymbol{\gamma}^\top$.
- Repeat the following steps till convergence:
  1. Update primal variable $\boldsymbol{T}$:

$$\begin{aligned}
&\boldsymbol{T}^{(m+1)} = \arg\min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu},\cdot)} \langle \boldsymbol{D}, \boldsymbol{T}\rangle + \langle \boldsymbol{Z}^{(m)}, \boldsymbol{T}\rangle + \epsilon \mathsf{KL}(\boldsymbol{T}\|\boldsymbol{S}^{(m)}) \\
&\xrightarrow{\text{First-order optimality}} \boldsymbol{T}^{(m+1)} = \exp\Big(\frac{\epsilon\log \boldsymbol{S}^{(m)} - \boldsymbol{D} - \boldsymbol{Z}^{(m)}}{\epsilon}\Big) \\
&\xrightarrow{\text{Project to } \Pi(\boldsymbol{\mu},\cdot)} \boldsymbol{T}^{(m+1)} = \mathsf{diag}(\boldsymbol{\mu})\mathsf{Softmax}_r\Big(\frac{\epsilon\log \boldsymbol{S}^{(m)} - \boldsymbol{D} - \boldsymbol{Z}^{(m)}}{\epsilon}\Big).
\end{aligned} \tag{14}$$

  2. Update auxiliary variable $\boldsymbol{S}$:

$$\begin{aligned}
\boldsymbol{S}^{(m+1)} &= \arg\min_{\boldsymbol{S}\in\Pi(\cdot,\boldsymbol{\gamma})} -\langle \boldsymbol{Z}^{(m)}, \boldsymbol{S}\rangle + \epsilon \mathsf{KL}(\boldsymbol{S}\|\boldsymbol{T}^{(m+1)}) \\
&= \mathsf{Softmax}_c\Big(\frac{\epsilon\log \boldsymbol{T}^{(m+1)} + \boldsymbol{Z}^{(m)}}{\epsilon}\Big)\mathsf{diag}(\boldsymbol{\gamma}).
\end{aligned}$$

Bregman ADMM: Solve OT without Sinkhorn

- Initialize $\boldsymbol{Z}^{(0)} = \mathbf{0}$, $\boldsymbol{T}^{(0)} = \boldsymbol{S}^{(0)} = \boldsymbol{\mu}\boldsymbol{\gamma}^\top$.
- Repeat the following steps till convergence:
  1. Update primal variable $\boldsymbol{T}$:

$$\begin{aligned}
&\boldsymbol{T}^{(m+1)} = \arg\min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu},\cdot)} \langle \boldsymbol{D}, \boldsymbol{T}\rangle + \langle \boldsymbol{Z}^{(m)}, \boldsymbol{T}\rangle + \epsilon \mathsf{KL}(\boldsymbol{T}\|\boldsymbol{S}^{(m)}) \\
&\xrightarrow{\text{First-order optimality}} \boldsymbol{T}^{(m+1)} = \exp\Big(\frac{\epsilon\log \boldsymbol{S}^{(m)} - \boldsymbol{D} - \boldsymbol{Z}^{(m)}}{\epsilon}\Big) \\
&\xrightarrow{\text{Project to } \Pi(\boldsymbol{\mu},\cdot)} \boldsymbol{T}^{(m+1)} = \mathsf{diag}(\boldsymbol{\mu})\mathsf{Softmax}_r\Big(\frac{\epsilon\log \boldsymbol{S}^{(m)} - \boldsymbol{D} - \boldsymbol{Z}^{(m)}}{\epsilon}\Big).
\end{aligned} \tag{14}$$

  2. Update auxiliary variable $\boldsymbol{S}$:

$$\begin{aligned}
\boldsymbol{S}^{(m+1)} &= \arg\min_{\boldsymbol{S}\in\Pi(\cdot,\boldsymbol{\gamma})} -\langle \boldsymbol{Z}^{(m)}, \boldsymbol{S}\rangle + \epsilon \mathsf{KL}(\boldsymbol{S}\|\boldsymbol{T}^{(m+1)}) \\
&= \mathsf{Softmax}_c\Big(\frac{\epsilon\log \boldsymbol{T}^{(m+1)} + \boldsymbol{Z}^{(m)}}{\epsilon}\Big)\mathsf{diag}(\boldsymbol{\gamma}).
\end{aligned} \tag{15}$$

  3. Update dual variable $\boldsymbol{Z}$:

$$\boldsymbol{Z}^{(m+1)} = \boldsymbol{Z}^{(m)} + \epsilon(\boldsymbol{T}^{(m+1)} - \boldsymbol{S}^{(m+1)}). \tag{16}$$

Impose Structures on OT Plans: Low-rank Optimal Transport

- Low-rank OT plans [Scetbon et al. 2021]: $\boldsymbol{T} = \boldsymbol{Q}\text{diag}^{-1}(\boldsymbol{g})\boldsymbol{R}^\top \in \Pi(\boldsymbol{\mu}, \boldsymbol{\gamma})$

$$\begin{aligned} \min_{\boldsymbol{Q},\boldsymbol{R},\boldsymbol{g}} \ & \langle \boldsymbol{D}, \boldsymbol{Q}\text{diag}^{-1}(\boldsymbol{g})\boldsymbol{R}^\top \rangle, \\ s.t. \ & \boldsymbol{Q} \in \mathbb{R}_+^{N\times r}, \ \boldsymbol{Q}\mathbf{1}_r = \boldsymbol{\mu}, \ \boldsymbol{R} \in \mathbb{R}_+^{M\times r}, \ \boldsymbol{R}\mathbf{1}_r = \boldsymbol{\gamma}, \\ & \boldsymbol{g} \in \mathbb{R}_+^r, \ \boldsymbol{R}^\top \mathbf{1}_N = \boldsymbol{Q}^\top \mathbf{1}_M = \boldsymbol{g}. \end{aligned} \tag{17}$$

Impose Structures on OT Plans: Low-rank Optimal Transport

- Low-rank OT plans [Scetbon et al. 2021]: $\boldsymbol{T} = \boldsymbol{Q}\text{diag}^{-1}(\boldsymbol{g})\boldsymbol{R}^\top \in \Pi(\boldsymbol{\mu}, \boldsymbol{\gamma})$

$$\begin{aligned} &\min_{\boldsymbol{Q},\boldsymbol{R},\boldsymbol{g}} \langle \boldsymbol{D}, \boldsymbol{Q}\text{diag}^{-1}(\boldsymbol{g})\boldsymbol{R}^\top \rangle, \\ &s.t.\ \boldsymbol{Q} \in \mathbb{R}_+^{N\times r},\ \boldsymbol{Q}\mathbf{1}_r = \boldsymbol{\mu},\ \boldsymbol{R} \in \mathbb{R}_+^{M\times r},\ \boldsymbol{R}\mathbf{1}_r = \boldsymbol{\gamma}, \\ &\boldsymbol{g} \in \mathbb{R}_+^r,\ \boldsymbol{R}^\top \mathbf{1}_N = \boldsymbol{Q}^\top \mathbf{1}_M = \boldsymbol{g}. \end{aligned} \tag{17}$$

- A mirror descent scheme w.r.t. the KL-divergence, leading to the Dykstra's Algorithm in each step: In the m-th step:

$$\begin{aligned} &\boldsymbol{Q}^{(m+1)}, \boldsymbol{R}^{(m+1)}, \boldsymbol{g}^{(m+1)} = \arg\min_{\boldsymbol{Q},\boldsymbol{R},\boldsymbol{g}\in\Omega} \text{KL}(\{\boldsymbol{Q},\boldsymbol{R},\boldsymbol{g}\} \| \{\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \boldsymbol{\xi}_3\}), \\ &\boldsymbol{\xi}_1 = \boldsymbol{Q}^{(m)} \odot \exp(-\epsilon_m \boldsymbol{D}\boldsymbol{R}^{(m)}\text{diag}^{-1}(\boldsymbol{g}^{(m)})), \\ &\boldsymbol{\xi}_2 = \boldsymbol{R}^{(m)} \odot \exp(-\epsilon_m \boldsymbol{D}^\top \boldsymbol{Q}^{(m)}\text{diag}^{-1}(\boldsymbol{g}^{(m)})), \\ &\boldsymbol{\xi}_3 = \boldsymbol{g}^{(m)} \odot \exp(\epsilon_m \text{diag}((\boldsymbol{Q}^{(m)})^\top \boldsymbol{D}\boldsymbol{R}^{(m)})/(\boldsymbol{g}^{(m)})^2). \end{aligned} \tag{18}$$

[Scetbon et al. 2021] Meyer Scetbon, Marco Cuturi, and Gabriel Peyré. Low-rank sinkhorn factorization, ICML, 2021.

Impose Structures on OT Plans: Sparse Optimal Transport

- **Replace the entropic regularizer to a quadratic regularizer** [Blondel et al. 2018]:

$$\min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu},\boldsymbol{\gamma})} \langle \boldsymbol{D}, \boldsymbol{T}\rangle + \frac{\epsilon}{2}\|\boldsymbol{T}\|_F^2. \tag{19}$$

Impose Structures on OT Plans: Sparse Optimal Transport

- **Replace the entropic regularizer to a quadratic regularizer** [Blondel et al. 2018]:

$$\min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu},\boldsymbol{\gamma})} \langle \boldsymbol{D}, \boldsymbol{T}\rangle + \frac{\epsilon}{2}\|\boldsymbol{T}\|_F^2. \tag{19}$$

- Applying the L-BFGS algorithm to solve the smoothed dual formulation of (19), the OT plan has a closed-form expression: for $\boldsymbol{T}^* = [t^*_{mn}]$,

$$t^*_{mn} = \frac{1}{\epsilon}[a^*_m + b^*_n - d_{mn}]_+.$$

Impose Structures on OT Plans: Sparse Optimal Transport

- **Replace the entropic regularizer to a quadratic regularizer** [Blondel et al. 2018]:

$$\min_{\boldsymbol{T}\in\Pi(\boldsymbol{\mu},\boldsymbol{\gamma})} \langle \boldsymbol{D}, \boldsymbol{T}\rangle + \frac{\epsilon}{2}\|\boldsymbol{T}\|_F^2. \tag{19}$$

- Applying the L-BFGS algorithm to solve the smoothed dual formulation of (19), the OT plan has a closed-form expression: for $\boldsymbol{T}^* = [t^*_{mn}]$,

$$t^*_{mn} = \frac{1}{\epsilon}[a^*_m + b^*_n - d_{mn}]_+. \tag{20}$$

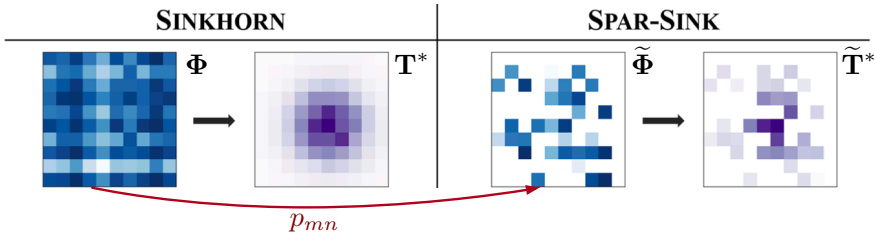
- This problem is highly correlated with LASSO, leading to a sparse OT plan.

[Blondel et al. 2018] Blondel, Mathieu, Vivien Seguy, and Antoine Rolet. Smooth and sparse optimal transport. AISTATS, 2018.

Impose Structures on OT Plans: Sparse Optimal Transport

- **Sample the OT plan randomly via importance sparsification** [Li et al. 2023]: apply the principle of Poisson sampling to sketch the kernel matrix $\mathbf{\Phi} = [\phi_{mn}]$ to s nonzero elements:

$$\widetilde{\mathbf{\Phi}} = [\tilde{\phi}_{mn}], \quad \text{where } \tilde{\phi}_{mn} = \begin{cases} \frac{\phi_{mn}}{p^*_{mn}} & \text{with prob.} p^*_{mn} = \min\{1, p_{mn} s\} \\ 0 & \text{otherwise.} \end{cases} \tag{21}$$



[Li et al. 2023] Li, M., Yu, J., Li, T. and Meng, C., Importance Sparsification for Sinkhorn Algorithm. JMLR, 2023.

Impose Structures on OT Plans: Sparse Optimal Transport

- The sampling probability $\boldsymbol{P} = [p_{mn}]$ is determined by the upper bound of $\boldsymbol{D} \odot \boldsymbol{T}^*$:

$$d_{mn} \leq c_0,\ t^*_{mn} \leq \mu_m, \nu_n \ \Rightarrow\ d_{mn} t^*_{mn} \leq c_0 \sqrt{\mu_m \nu_n} \ \Rightarrow\ p_{mn} = \frac{\sqrt{\mu_m \nu_n}}{\sum_{m,n} \sqrt{\mu_m \nu_n}} \quad (22)$$

Impose Structures on OT Plans: Sparse Optimal Transport

- The sampling probability $\boldsymbol{P} = [p_{mn}]$ is determined by the upper bound of $\boldsymbol{D} \odot \boldsymbol{T}^*$:

$$d_{mn} \leq c_0,\ t^*_{mn} \leq \mu_m, \nu_n \ \Rightarrow\ d_{mn} t^*_{mn} \leq c_0 \sqrt{\mu_m \nu_n} \ \Rightarrow\ p_{mn} = \frac{\sqrt{\mu_m \nu_n}}{\sum_{m,n} \sqrt{\mu_m \nu_n}} \quad (22)$$

- Reduce the complexity from $\mathcal{O}(N^2)$ to $\mathcal{O}(N \log N)$ when $s \approx N \log N$.

Impose Structures on OT Plans: Sparse Optimal Transport

- The sampling probability $\boldsymbol{P} = [p_{mn}]$ is determined by the upper bound of $\boldsymbol{D} \odot \boldsymbol{T}^*$:

$$d_{mn} \leq c_0,\ t^*_{mn} \leq \mu_m, \nu_n \ \Rightarrow\ d_{mn} t^*_{mn} \leq c_0 \sqrt{\mu_m \nu_n} \ \Rightarrow\ p_{mn} = \frac{\sqrt{\mu_m \nu_n}}{\sum_{m,n} \sqrt{\mu_m \nu_n}} \quad (22)$$

- Reduce the complexity from $\mathcal{O}(N^2)$ to $\mathcal{O}(N \log N)$ when $s \approx N \log N$.
- The approximation error between $\widehat{W}_{p,\epsilon}$ and $\widetilde{W}_{p,\epsilon}$ is bounded:

$$|\widehat{W}_{p,\epsilon} - \widetilde{W}_{p,\epsilon}| \leq c\epsilon \sqrt{\frac{N^{3-2\alpha}}{s}},\ \text{where } c > 0,\ \alpha \in (0.5, 1). \quad (23)$$

Sliced Wasserstein: A Surrogate of Wasserstein Distance

Motivation: The optimal transport between 1D distributions is relatively easy to solve.

Sliced Wasserstein: A Surrogate of Wasserstein Distance

Motivation: The optimal transport between 1D distributions is relatively easy to solve.

- When $\dim(\mathcal{X}) = 1$, W_p has a closed form, related to **1D histogram transform and equalization**.

$$W_p(\mu, \gamma) = \left(\int_0^1 |F^{-1}(z) - G^{-1}(z)|^p \mathrm{d}z \right)^{1/p}, \tag{24}$$

where $F, G : \mathcal{X} \mapsto [0, 1]$ are CDF's of μ and ν.

Sliced Wasserstein: A Surrogate of Wasserstein Distance

Motivation: The optimal transport between 1D distributions is relatively easy to solve.

- When $\dim(\mathcal{X}) = 1$, W_p has a closed form, related to **1D histogram transform and equalization**.

$$W_p(\mu, \gamma) = \Big(\int_0^1 |F^{-1}(z) - G^{-1}(z)|^p \mathrm{d}z \Big)^{1/p}, \tag{24}$$

  where $F, G : \mathcal{X} \mapsto [0, 1]$ are CDF's of μ and ν.

- Given $\boldsymbol{x} = \{x_n\}_{n=1}^N \sim \mu$ and $\boldsymbol{y} = \{y_n\}_{n=1}^N \sim \nu$:

$$\widehat{W}_p(\boldsymbol{x}, \boldsymbol{y}) = \Big(\sum_{n=1}^N |x_n - y_{\sigma(n)}|^p \mathrm{d}z \Big)^{1/p}, \tag{25}$$

  where σ denotes the sorting operation.

Theorem: For one dimensional $x_1 \leq ... \leq x_N$ and $y_1 \leq ... \leq y_N$, identity permutation ($\sigma(n) = n$ for $n = 1, ..., N$) leads to the optimal transport between them.

Sliced Wasserstein: A Surrogate of Wasserstein Distance

Sliced-Wasserstein distance [Bonneel et al., 2015]:

- Given two distributions μ and γ defined on a metric space $(\mathcal{X} \subset \mathbb{R}^D, d_X)$, projecting $\forall x \in \mathcal{X}$ through a linear projection $\theta \in \mathcal{S}^{D-1}$, *i.e.*, $R_\theta(x) = \langle x, \theta \rangle$, leads to
  - A 1D metric space $(R_\theta(\mathcal{X}), d_{R_\theta(\mathcal{X})})$.
  - The one-dimensional distributions after projection $R_{\theta\#}\mu$ and $R_{\theta\#}\gamma$.

Sliced Wasserstein: A Surrogate of Wasserstein Distance

Sliced-Wasserstein distance [Bonneel et al., 2015]:

- Given two distributions μ and γ defined on a metric space $(\mathcal{X} \subset \mathbb{R}^D, d_X)$, projecting $\forall x \in \mathcal{X}$ through a linear projection $\theta \in \mathcal{S}^{D-1}$, *i.e.*, $R_\theta(x) = \langle x, \theta \rangle$, leads to
  - A 1D metric space $(R_\theta(\mathcal{X}), d_{R_\theta(\mathcal{X})})$.
  - The one-dimensional distributions after projection $R_{\theta\#}\mu$ and $R_{\theta\#}\gamma$.

$$SW_p(\mu, \gamma) := \mathbb{E}_{\theta \sim p_{\mathcal{S}^{D-1}}}[W_p(R_{\theta\#}\mu, R_{\theta\#}\gamma)] = \int_{\theta \in \mathcal{S}^{D-1}} W_p(R_{\theta\#}\mu, R_{\theta\#}\gamma) \mathrm{d}p(\theta) \quad (26)$$

[Bonneel et al., 2015] Bonneel, N., Rabin, J., Peyré, G. and Pfister, H., Sliced and radon wasserstein barycenters of measures. Journal of Mathematical Imaging and Vision, 2015.

Sliced Wasserstein: A Surrogate of Wasserstein Distance

- Practical implementation:
  - Samples $\boldsymbol{X} = \{x_n\}_{n=1}^N \sim \mu$ and $\boldsymbol{Y} = \{y_n\}_{n=1}^N \sim \gamma$ are provided.
  - Finite number of projections are sampled based on a distribution $\{\theta_l\}_{l=1}^L \sim p_{\mathcal{S}^{D-1}}$.

Sliced Wasserstein: A Surrogate of Wasserstein Distance

- Practical implementation:
  - Samples $\boldsymbol{X} = \{x_n\}_{n=1}^N \sim \mu$ and $\boldsymbol{Y} = \{y_n\}_{n=1}^N \sim \gamma$ are provided.
  - Finite number of projections are sampled based on a distribution $\{\theta_l\}_{l=1}^L \sim p_{\mathcal{S}^{D-1}}$.
- Sample-based sliced Wasserstein distance:

$$\widehat{SW}_p(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{L} \sum_{l=1}^{L} \left(\min_{\boldsymbol{T} \in \Pi(\frac{1}{N}\mathbf{1}_N, \frac{1}{N}\mathbf{1}_N)} \sum_{m,n=1}^{N} |\theta_l^\top x_m - \theta_l^\top y_n|^p t_{mn} \right)^{1/p}$$

Sliced Wasserstein: A Surrogate of Wasserstein Distance

- Practical implementation:
  - Samples $\boldsymbol{X} = \{x_n\}_{n=1}^N \sim \mu$ and $\boldsymbol{Y} = \{y_n\}_{n=1}^N \sim \gamma$ are provided.
  - Finite number of projections are sampled based on a distribution $\{\theta_l\}_{l=1}^L \sim p_{\mathcal{S}^{D-1}}$.
- Sample-based sliced Wasserstein distance:

$$\begin{aligned}
\widehat{SW}_p(\boldsymbol{X}, \boldsymbol{Y}) &= \frac{1}{L}\sum_{l=1}^{L}\left(\min_{\boldsymbol{T}\in\Pi(\frac{1}{N}\mathbf{1}_N, \frac{1}{N}\mathbf{1}_N)} \sum_{m,n=1}^{N} |\theta_l^\top x_m - \theta_l^\top y_n|^p t_{mn}\right)^{1/p} \\
&= \frac{1}{L}\sum_{l=1}^{L}\left(\frac{1}{N}\min_{\sigma\in\mathcal{P}_N} \sum_{n=1}^{N} |\theta_l^\top x_n - \theta_l^\top y_{\sigma(n)}|^p\right)^{1/p} \\
&= \frac{1}{L}\sum_{l=1}^{L}\left(\frac{1}{N}\sum_{n=1}^{N} |\theta_l^\top x_{\sigma^*(n)} - \theta_l^\top y_{\sigma^*(n)}|^p\right)^{1/p}
\end{aligned} \tag{27}$$

Extensions of Sliced Wasserstein

Max-sliced Wasserstein (MSW) [Deshpande et al., 2019]: Instead of randomly sampling projections, learn the optimal one in an adversarial way.

- Given two distributions μ and γ defined on a metric space $(\mathcal{X} \subset \mathbb{R}^D, d_X)$, find the optimal projection that maximizes the 1D Wasserstein distance:

$$MSW_p(\mu, \gamma) := \max_{\theta \in \mathcal{S}^{D-1}} W_p(R_{\theta\#}\mu, R_{\theta\#}\gamma) \tag{28}$$

Extensions of Sliced Wasserstein

Max-sliced Wasserstein (MSW) [Deshpande et al., 2019]: Instead of randomly sampling projections, learn the optimal one in an adversarial way.

- Given two distributions μ and γ defined on a metric space $(\mathcal{X} \subset \mathbb{R}^D, d_X)$, find the optimal projection that maximizes the 1D Wasserstein distance:

$$MSW_p(\mu, \gamma) := \max_{\theta \in \mathcal{S}^{D-1}} W_p(R_{\theta\#}\mu, R_{\theta\#}\gamma) \tag{28}$$

- Given samples $\boldsymbol{X} = \{x_n\}_{n=1}^N \sim \mu$ and $\boldsymbol{Y} = \{y_n\}_{n=1}^N \sim \gamma$,

$$\widehat{MSW}_p(\boldsymbol{X}, \boldsymbol{Y}) := \max_{\theta \in \mathcal{S}^{D-1}} \left(\min_{\sigma \in \mathcal{P}_N} \sum_{n=1}^{N} |\theta^\top x_n - \theta^\top y_{\sigma(n)}|^p \right)^{1/p} \tag{29}$$

[Deshpande et al., 2019] Deshpande, I., Hu, Y.T., Sun, R., Pyrros, A., Siddiqui, N., Koyejo, S., Zhao, Z., Forsyth, D. and Schwing, A.G., Max-sliced wasserstein distance and its use for gans. CVPR, 2019.

Extensions of Sliced Wasserstein

- MSW_p is strongly equivalence to W_p [Paty et al. 2019, Bayraktar et al. 2021]: for $p = 1, 2$,

$$\exists 0 < c_1 < c_2,\ c_1 MSW_p \leq W_p \leq c_2 MSW_p. \tag{30}$$

- In other words, in many situations, using MSW_p should be comparable to using W_p **concerning distance metric**.

[Paty et al. 2019] Paty, F.P. and Cuturi, M., Subspace robust Wasserstein distances. ICML, 2019.
[Bayraktar et al. 2021] Bayraktar, E. and Guo, G., Strong equivalence between metrics of Wasserstein type. 2021.

Extensions of Sliced Wasserstein

Generalized sliced Wasserstein (GSW) [Kolouri, et al., 2019]: Replacing the linear projections to nonlinear ones (by generalized Radon transformation)

- Given two distributions μ and γ defined on a metric space $(\mathcal{X} \subset \mathbb{R}^D, d_X)$, we have

$$\begin{aligned} GSW_p(\mu, \gamma) &:= \int_{F_\theta \in \Omega} W_p(F_{\theta\#}\mu, F_{\theta\#}\gamma) dp(\theta), \\ MGSW_p(\mu, \gamma) &:= \max_{F_\theta \in \Omega} W_p(F_{\theta\#}\mu, F_{\theta\#}\gamma) \end{aligned} \tag{31}$$

where $F_\theta \in \Omega$ is the generalized Radon transformation and θ is rotation angle.

Extensions of Sliced Wasserstein

Generalized sliced Wasserstein (GSW) [Kolouri, et al., 2019]: Replacing the linear projections to nonlinear ones (by generalized Radon transformation)

- Given two distributions μ and γ defined on a metric space $(\mathcal{X} \subset \mathbb{R}^D, d_X)$, we have

$$\begin{aligned} GSW_p(\mu, \gamma) &:= \int_{F_\theta \in \Omega} W_p(F_{\theta\#}\mu, F_{\theta\#}\gamma) dp(\theta), \\ MGSW_p(\mu, \gamma) &:= \max_{F_\theta \in \Omega} W_p(F_{\theta\#}\mu, F_{\theta\#}\gamma) \end{aligned} \tag{31}$$

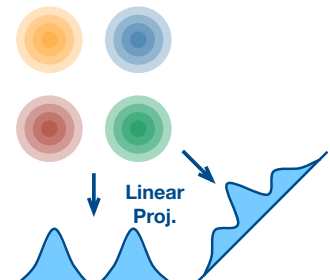
  where $F_\theta \in \Omega$ is the generalized Radon transformation and θ is rotation angle.

- Alternating optimization is applied to compute these variants.

[Kolouri, et al., 2019] Kolouri, S., Nadjahi, K., Simsekli, U., Badeau, R., & Rohde, G. Generalized sliced wasserstein distances. NeurIPS, 2019.

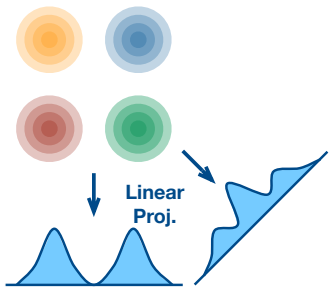
Hilbert Curve Projection Distance: Locality-preserving Projection + OT

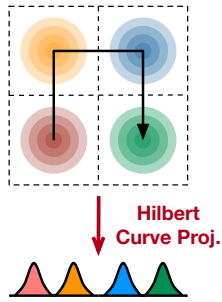
Motivation: Linear projections used in SW often break the **locality-preserving property**.



Hilbert Curve Projection Distance: Locality-preserving Projection + OT

Motivation: Linear projections used in SW often break the **locality-preserving property**.

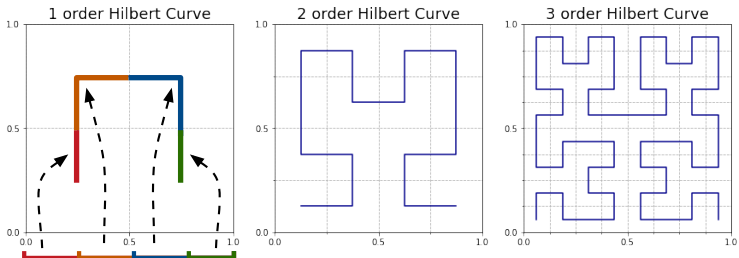




Hilbert curve, a special kind of space-filling curve, provides us a potential projection method that has locality-preserving property.

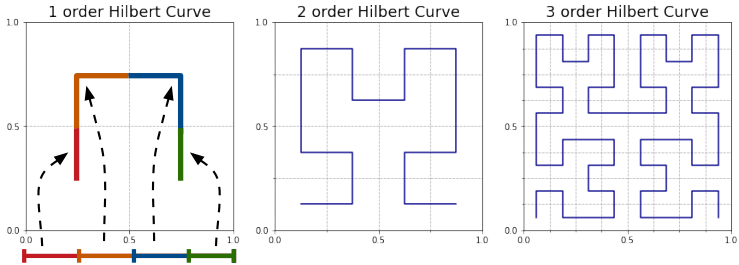
Hilbert Curve Projection Distance: Locality-preserving Projection + OT

- A K-order Hilbert curve H_K:
  - Partition the $[0,1]$ and D-dimensional unit hyper-cube $[0,1]^D$ into $(2^K)^D$ parts.
  - **Construct a bijection between them**.



Hilbert Curve Projection Distance: Locality-preserving Projection + OT

- A K-order Hilbert curve H_K:
  - Partition the $[0,1]$ and D-dimensional unit hyper-cube $[0,1]^D$ into $(2^K)^D$ parts.
  - **Construct a bijection between them**.



- $H(x) = \lim_{K\to\infty} H_K(x)$ is a surjection $H : [0,1] \to [0,1]^d$ (space-filing curve).
- H covers the entire hyper-cube and enjoys the **locality-preserving property**:

$$\|H(x) - H(y)\|_2 \leq 2\sqrt{d+3}|x-y|^{1/d}, \ \forall x, y \in [0,1]. \tag{32}$$

Hilbert Curve Projection Distance: Locality-preserving Projection + OT

- Given a probability measure μ defined on a hyper-cube Ω_μ,
  - **The Hilbert curve:** $H_\mu : [0,1] \mapsto \Omega_\mu$
  - **The CDF along H_μ, denoted as** $g : [0,1] \mapsto [0,1]$**:**

$$g_\mu(t) = \inf_{s\in[0,t]} \mu\Big(\underbrace{H_\mu([0,s])}_{\text{A Borel set in } \Omega_\mu} \Big). \tag{33}$$

Hilbert Curve Projection Distance: Locality-preserving Projection + OT

- Given a probability measure μ defined on a hyper-cube Ω_μ,
  - **The Hilbert curve:** $H_\mu : [0,1] \mapsto \Omega_\mu$
  - **The CDF along** H_μ**, denoted as** $g : [0,1] \mapsto [0,1]$**:**

$$g_\mu(t) = \inf_{s\in[0,t]} \mu\Big(\underbrace{H_\mu([0,s])}_{\text{A Borel set in } \Omega_\mu} \Big). \tag{33}$$

- The Hilbert Curve Projection (HCP) distance [Li, et al. 2022] is

$$HCP_p(\mu,\gamma) := \Big(\int_0^1 \| H_\mu(g_\mu^{-1}(t)) - H_\gamma(g_\gamma^{-1}(t)) \|_p^p \mathrm{d}t \Big)^{\frac{1}{p}}$$
$$\text{Compare to 1D } W_p(\mu,\gamma) = \Big(\int_0^1 |F^{-1}(z) - G^{-1}(z)|^p \mathrm{d}z \Big)^{1/p} \tag{34}$$

Hilbert Curve Projection Distance: Locality-preserving Projection + OT

- Given a probability measure μ defined on a hyper-cube Ω_μ,
  - **The Hilbert curve:** $H_\mu : [0,1] \mapsto \Omega_\mu$
  - **The CDF along H_μ, denoted as** $g : [0,1] \mapsto [0,1]$**:**

$$g_\mu(t) = \inf_{s\in[0,t]} \mu\Big(\underbrace{H_\mu([0,s])}_{\text{A Borel set in } \Omega_\mu} \Big). \tag{33}$$

- The Hilbert Curve Projection (HCP) distance [Li, et al. 2022] is

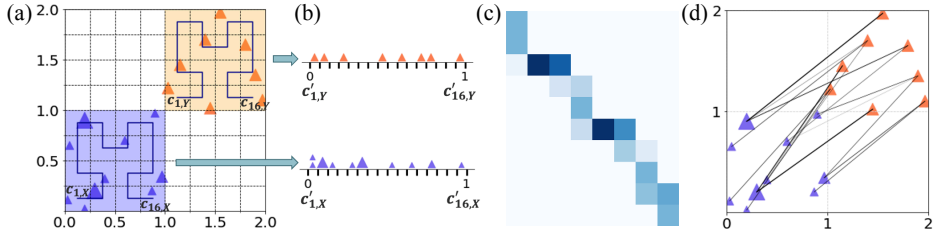
$$HCP_p(\mu,\gamma) := \Big(\int_0^1 \| H_\mu(g_\mu^{-1}(t)) - H_\gamma(g_\gamma^{-1}(t)) \|_p^p \mathrm{d}t \Big)^{\frac{1}{p}}$$
$$\text{Compare to 1D } W_p(\mu,\gamma) = \Big(\int_0^1 |F^{-1}(z) - G^{-1}(z)|^p \mathrm{d}z \Big)^{1/p} \tag{34}$$

- **A valid metric for probability measures + An upper bound of W_p:**

$$HCP_p(\mu_N,\mu) \to 0, \quad \text{and} \quad W_p(\mu,\gamma) \le HCP_p(\mu,\gamma). \tag{35}$$

[Li et al, 2022] Li, T., Meng, C., Xu, H. and Yu, J., Hilbert curve projection distance for distribution comparison. arXiv:2205.15059, 2022.

Hilbert Curve Projection Distance: Locality-preserving Projection + OT



1. Project D-dimensional samples along their K-order Hilbert curves, and determine the OT plan accordingly. ($\mathcal{O}((N+M)DK)$)
2. Determine the OT plan via sorting the projected samples. ($\mathcal{O}(N \log N + M \log M)$)
3. Compute the HCP distance by the raw samples and the OT plan.

$$\widehat{HCP}_p(\boldsymbol{X}, \boldsymbol{Y}) = \left(\sum_{m,n} \|\boldsymbol{x}_m - \boldsymbol{y}_n\|_p^p t^*_{mn} \right)^{1/p}. \tag{36}$$

Summary

- W_p and its variants (e.g., SW_p, MSW_p HCP_p, and so on) provide valid metrics for probability measures.
  - MSW_p is strongly equivalent to W_p
  - SW_p is weakly equivalent to W_p
  - HCP_p is an upper bound of W_p

Summary

- W_p and its variants (e.g., SW_p, MSW_p HCP_p, and so on) provide valid metrics for probability measures.
  - MSW_p is strongly equivalent to W_p
  - SW_p is weakly equivalent to W_p
  - HCP_p is an upper bound of W_p
- Efficient approximation methods (Sinkhorn, Proximal Point, Bregman ADMM, etc.) are proposed with the help of various smoothness regularizers.
  - Sublinear convergence rate (i.e., $\mathcal{O}(1/\epsilon^2)$ steps to achieve ϵ-approximation)
  - Reduce the complexity to $\mathcal{O}(N^2)$

Summary

- W_p and its variants (e.g., SW_p, MSW_p HCP_p, and so on) provide valid metrics for probability measures.
  - MSW_p is strongly equivalent to W_p
  - SW_p is weakly equivalent to W_p
  - HCP_p is an upper bound of W_p
- Efficient approximation methods (Sinkhorn, Proximal Point, Bregman ADMM, etc.) are proposed with the help of various smoothness regularizers.
  - Sublinear convergence rate (i.e., $\mathcal{O}(1/\epsilon^2)$ steps to achieve ϵ-approximation)
  - Reduce the complexity to $\mathcal{O}(N^2)$
- Structured OT plans (Low-rank and/or sparse OT plans) often lead to further accelerations.
  - The time complexity of Low-rank OT is $\mathcal{O}(N^2 r)$ but it reduces memory cost significantly.
  - Apply importance sparsification reduces the complexity to $\mathcal{O}(N \log N)$

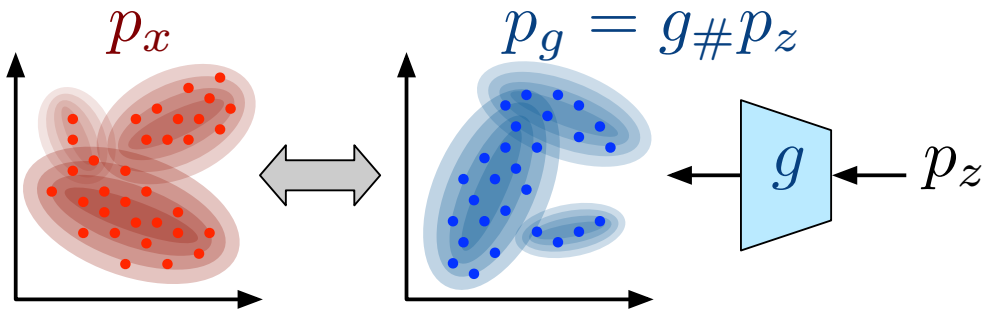
Summary

- W_p and its variants (e.g., SW_p, MSW_p HCP_p, and so on) provide valid metrics for probability measures.
  - MSW_p is strongly equivalent to W_p
  - SW_p is weakly equivalent to W_p
  - HCP_p is an upper bound of W_p
- Efficient approximation methods (Sinkhorn, Proximal Point, Bregman ADMM, etc.) are proposed with the help of various smoothness regularizers.
  - Sublinear convergence rate (i.e., $\mathcal{O}(1/\epsilon^2)$ steps to achieve ϵ-approximation)
  - Reduce the complexity to $\mathcal{O}(N^2)$
- Structured OT plans (Low-rank and/or sparse OT plans) often lead to further accelerations.
  - The time complexity of Low-rank OT is $\mathcal{O}(N^2 r)$ but it reduces memory cost significantly.
  - Apply importance sparsification reduces the complexity to $\mathcal{O}(N \log N)$
- Potential applications:
  - For distance-centric applications: design loss functions.
  - For OT plan-centric applications: solve matching problems and design models.

5-min break for Q & A

Outline

Generative Modeling = Distribution Fitting and Matching



- $g : \mathcal{Z} \mapsto \mathcal{X}$ is the generator/decoder.
- p_z is the (predefined) latent distribution, and $p_g = g_{\#}p_z$ is the model distribution.
- Learn g to fit data distribution p_x by p_g.

OT-based Generative Modeling Paradigms

Solution 1: Minimize W_1 approximately in its dual-form.

- WGAN and its variants

OT-based Generative Modeling Paradigms

Solution 1: Minimize W_1 approximately in its dual-form.

- WGAN and its variants

Solution 2: Minimize W_2 approximately in its primal-form.

- Differentiable Sinkhorn divergence (SinkDiff) and its variants
- Wasserstein Autoencoder (WAE) and its variants

OT-based Generative Modeling Paradigms

Solution 1: Minimize W_1 approximately in its dual-form.

- WGAN and its variants

Solution 2: Minimize W_2 approximately in its primal-form.

- Differentiable Sinkhorn divergence (SinkDiff) and its variants
- Wasserstein Autoencoder (WAE) and its variants

Solution 3: Minimize efficient surrogate of W_2, e.g., SW_2 and MSW_2

- Sliced Wasserstein generative (SGW) model and its variants.

The Dual Form of W_p

The primal form of W_p:

$$W_p^p(\mu,\gamma) = \inf_{\pi\in\Pi(\mu,\gamma)} \int_{x,y\in\mathcal{X}^2} \|x-y\|_p^p \mathrm{d}\pi(x,y) = \inf_{\pi\in\Pi(\mu,\gamma)} \mathbb{E}_{(x,y)\sim\pi}[\|x-y\|_p^p] \quad (37)$$

The Dual Form of W_p

The primal form of W_p:

$$W_p^p(\mu,\gamma) = \inf_{\pi\in\Pi(\mu,\gamma)} \int_{x,y\in\mathcal{X}^2} \|x-y\|_p^p \mathrm{d}\pi(x,y) = \inf_{\pi\in\Pi(\mu,\gamma)} \mathbb{E}_{(x,y)\sim\pi}[\|x-y\|_p^p] \quad (37)$$

The dual form of W_p:

$$W_p^p(\mu,\gamma) = \sup_{\phi,\psi} \int_{x\in\mathcal{X}} \phi(x)\mathrm{d}\mu(x) - \int_{y\in\mathcal{X}} \psi(y)\mathrm{d}\gamma(y) = \sup_{\phi,\psi} \mathbb{E}_{x\sim\mu}[\phi(x)] - \mathbb{E}_{y\sim\gamma}[\psi(y)]. \quad (38)$$

- $\phi(x) - \psi(y) \leq \|x-y\|_p^p$, $\forall\ x, y \in \mathcal{X}^2$.

The Dual Form of W_p

The primal form of W_p:

$$W_p^p(\mu,\gamma) = \inf_{\pi\in\Pi(\mu,\gamma)} \int_{x,y\in\mathcal{X}^2} \|x-y\|_p^p \mathrm{d}\pi(x,y) = \inf_{\pi\in\Pi(\mu,\gamma)} \mathbb{E}_{(x,y)\sim\pi}[\|x-y\|_p^p] \quad (37)$$

The dual form of W_p:

$$W_p^p(\mu,\gamma) = \sup_{\phi,\psi} \int_{x\in\mathcal{X}} \phi(x)\mathrm{d}\mu(x) - \int_{y\in\mathcal{X}} \psi(y)\mathrm{d}\gamma(y) = \sup_{\phi,\psi} \mathbb{E}_{x\sim\mu}[\phi(x)] - \mathbb{E}_{y\sim\gamma}[\psi(y)]. \quad (38)$$

- $\phi(x) - \psi(y) \leq \|x-y\|_p^p$, $\forall\ x,y \in \mathcal{X}^2$.

The dual form of W_1:

$$W_1(\mu,\gamma) = \sup_{f\in L_1} \int_{x\in\mathcal{X}} f(x)\mathrm{d}\mu(x) - \int_{y\in\mathcal{X}} f(y)\mathrm{d}\gamma(y) = \sup_{f\in L_1} \mathbb{E}_{x\sim\mu}[f(x)] - \mathbb{E}_{y\sim\gamma}[f(y)]. \quad (39)$$

- $f \in L_1$: f satisfies 1-Lipschitzness, i.e., $|f(x) - f(y)| \leq \|x-y\|_1$, $\forall\ x,y \in \mathcal{X}^2$.

Wasserstein Generative Adversarial Network (WGAN)

Wasserstein Generative Adversarial Network (WGAN) [Arjovsky et al., 2017]: Fit the model distribution p_g by minimizing its 1-Wasserstein distance to the data distribution p_x **in the dual-form**:

$$W_1(p_x, p_g) = \inf_{\pi \in \Pi(p_x, p_g)} \mathbb{E}_{(x,g(z))\sim\pi}[\|x - g(z)\|_1] = \sup_{f \in L_1} \mathbb{E}_x[f(x)] - \mathbb{E}_z[f(g(z))] \quad (40)$$

Wasserstein Generative Adversarial Network (WGAN)

Wasserstein Generative Adversarial Network (WGAN) [Arjovsky et al., 2017]: Fit the model distribution p_g by minimizing its 1-Wasserstein distance to the data distribution p_x **in the dual-form**:

$$W_1(p_x, p_g) = \inf_{\pi \in \Pi(p_x, p_g)} \mathbb{E}_{(x, g(z)) \sim \pi}[\|x - g(z)\|_1] = \sup_{f \in L_1} \mathbb{E}_x[f(x)] - \mathbb{E}_z[f(g(z))] \tag{40}$$

Therefore, we have

$$\inf_g W_1(p_x, p_g) \Longleftrightarrow \inf_g \sup_{f \in L_1} \mathbb{E}_x[f(x)] - \mathbb{E}_z[f(g(z))] \tag{41}$$

Wasserstein Generative Adversarial Network (WGAN)

Wasserstein Generative Adversarial Network (WGAN) [Arjovsky et al., 2017]: Fit the model distribution p_g by minimizing its 1-Wasserstein distance to the data distribution p_x **in the dual-form**:

$$W_1(p_x, p_g) = \inf_{\pi \in \Pi(p_x, p_g)} \mathbb{E}_{(x,g(z))\sim\pi}[\|x - g(z)\|_1] = \sup_{f \in L_1} \mathbb{E}_x[f(x)] - \mathbb{E}_z[f(g(z))] \quad (40)$$

Therefore, we have

$$\inf_g W_1(p_x, p_g) \Longleftrightarrow \inf_g \sup_{f \in L_1} \mathbb{E}_x[f(x)] - \mathbb{E}_z[f(g(z))] \quad (41)$$

Given a set of samples $X = \{x_n\}_{n=1}^N$ and a set of latent code $Z = \{z_n\}_{n=1}^N$, we have

$$\min_g \max_{f \in L_1} \sum_n [f(x_n)] - \sum_n [f(g(z_n))] \quad (42)$$

[Arjovsky et al., 2017] Arjovsky, M., Chintala, S. and Bottou, L., Wasserstein generative adversarial networks. ICML, 2017.

Improve WGAN with Gradient Penalty (WGAN-GP)

Motivations:

- We cannot hold the Lipschitzness of f strictly based on finite samples.
- WGAN clips f's weights to ensure its Lipchitzness, leading to undesired performance.

Improve WGAN with Gradient Penalty (WGAN-GP)

Motivations:

- We cannot hold the Lipschitzness of f strictly based on finite samples.
- WGAN clips f's weights to ensure its Lipchitzness, leading to undesired performance.

WGAN-GP [Gulrajani, et al., 2017]: Adding a gradient penalty leads to a soft but better Lipchitz regularizer.

$$\inf_g \sup_f \mathbb{E}_x[f(x)] - \mathbb{E}_z[f(g(z))] + \lambda \mathbb{E}_z[(\|\nabla_{g(z)} f(g(z))\| - 1)^2]. \tag{43}$$

Improve WGAN with Gradient Penalty (WGAN-GP)

Motivations:

- We cannot hold the Lipschitzness of f strictly based on finite samples.
- WGAN clips f's weights to ensure its Lipchitzness, leading to undesired performance.

WGAN-GP [Gulrajani, et al., 2017]: Adding a gradient penalty leads to a soft but better Lipchitz regularizer.

$$\inf_g \sup_f \mathbb{E}_x[f(x)] - \mathbb{E}_z[f(g(z))] + \lambda \mathbb{E}_z[(\|\nabla_{g(z)} f(g(z))\| - 1)^2]. \tag{43}$$

Given samples, we have

$$\min_g \max_f \sum_n f(x_n) - \sum_n f(g(z_n)) + \lambda \sum_n (\|\nabla_{g(z_n)} f(g(z_n))\| - 1)^2. \tag{44}$$

[Gulrajani, et al., 2017] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V. and Courville, A.C., Improved training of wasserstein gans. NeurIPS, 2017.

SinkDiff: Learning Generative Models Based on Sinkhorn Divergence

Motivations:

- Learn generative models by minimizing **the primal-form entropic optimal transport (EOT)**.

SinkDiff: Learning Generative Models Based on Sinkhorn Divergence

Motivations:

- Learn generative models by minimizing **the primal-form entropic optimal transport (EOT)**.

Problem 1: EOT is not a distance because of the entropic term (i.e., $W_{p,\epsilon}(\mu, \mu) \neq 0$).

SinkDiff: Learning Generative Models Based on Sinkhorn Divergence

Motivations:

- Learn generative models by minimizing **the primal-form entropic optimal transport (EOT)**.

Problem 1: EOT is not a distance because of the entropic term (i.e., $W_{p,\epsilon}(\mu, \mu) \neq 0$).

- **Sinkhorn Divergence** [Genevay et al., 2018]:

$$\bar{W}_{p,\epsilon}(\mu, \gamma) := 2W_{p,\epsilon}(\mu, \gamma) - W_{p,\epsilon}(\mu, \mu) - W_{p,\epsilon}(\gamma, \gamma). \tag{45}$$

SinkDiff: Learning Generative Models Based on Sinkhorn Divergence

Motivations:

- Learn generative models by minimizing **the primal-form entropic optimal transport (EOT)**.

Problem 1: EOT is not a distance because of the entropic term (i.e., $W_{p,\epsilon}(\mu,\mu) \neq 0$).

- **Sinkhorn Divergence** [Genevay et al., 2018]:

$$\bar{W}_{p,\epsilon}(\mu,\gamma) := 2W_{p,\epsilon}(\mu,\gamma) - W_{p,\epsilon}(\mu,\mu) - W_{p,\epsilon}(\gamma,\gamma). \tag{45}$$

  - $\epsilon \to 0$: $\bar{W}_{p,\epsilon}(\mu,\gamma) \to 2W_p(\mu,\gamma)$.
  - $\epsilon \to \infty$: $\bar{W}_{p,\epsilon}(\mu,\gamma) \to MMD(\mu,\gamma)$

SinkDiff: Learning Generative Models Based on Sinkhorn Divergence

Motivations:

- Learn generative models by minimizing **the primal-form entropic optimal transport (EOT)**.

Problem 1: EOT is not a distance because of the entropic term (i.e., $W_{p,\epsilon}(\mu,\mu) \neq 0$).

- **Sinkhorn Divergence** [Genevay et al., 2018]:

$$\bar{W}_{p,\epsilon}(\mu,\gamma) := 2W_{p,\epsilon}(\mu,\gamma) - W_{p,\epsilon}(\mu,\mu) - W_{p,\epsilon}(\gamma,\gamma). \tag{45}$$

- $\epsilon \to 0$: $\bar{W}_{p,\epsilon}(\mu,\gamma) \to 2W_p(\mu,\gamma)$.
- $\epsilon \to \infty$: $\bar{W}_{p,\epsilon}(\mu,\gamma) \to MMD(\mu,\gamma)$

SinkDiff Generative Model:

$$\inf_g \bar{W}_{2,\epsilon}(p_x, p_g) \approx \inf_{g,\pi\in\Pi_\epsilon(p_x,p_g)} \mathbb{E}_{x,g(z)\sim\pi}[\|x - g(z)\|_2^2] \tag{46}$$

[Genevay et al., 2018] Genevay, A., Peyré, G. and Cuturi, M., March. Learning generative models with sinkhorn divergences. AISTATS, 2018.

SinkDiff: Learning Generative Models Based on Sinkhorn Divergence

Problem 2: The distance between high-dimensional real and fake data suffers from the curse of dimensionality.

SinkDiff: Learning Generative Models Based on Sinkhorn Divergence

Problem 2: The distance between high-dimensional real and fake data suffers from the curse of dimensionality.

- Train an encoder f for dimensionality reduction in an adversarial way

$$\begin{aligned}&\sup_{f}\inf_{g}\bar{W}_{2,\epsilon}(f_{\#}p_x, f_{\#}p_g)\\&=\sup_{f}\inf_{g,\pi\in\Pi_\epsilon(f_{\#}p_x,f_{\#}p_g)}\mathbb{E}_{f(x),f(g(z))\sim\pi}[\|f(x)-f(g(z))\|_2^2]\end{aligned}\tag{47}$$

SinkDiff: Learning Generative Models Based on Sinkhorn Divergence

Problem 2: The distance between high-dimensional real and fake data suffers from the curse of dimensionality.

- Train an encoder f for dimensionality reduction in an adversarial way

$$\begin{aligned}&\sup_f \inf_g \bar{W}_{2,\epsilon}(f_{\#}p_x, f_{\#}p_g)\\&= \sup_f \inf_{g,\pi\in\Pi_\epsilon(f_{\#}p_x, f_{\#}p_g)} \mathbb{E}_{f(x),f(g(z))\sim\pi}[\|f(x)-f(g(z))\|_2^2]\end{aligned} \tag{47}$$

- Given a batch of samples,

$$\max_f \min_{g,\boldsymbol{T}\in\Pi_\epsilon} \langle \boldsymbol{D}(f(\boldsymbol{X}), f(g(\boldsymbol{Z}))), \boldsymbol{T}\rangle + \epsilon H(\boldsymbol{T}). \tag{48}$$

SinkDiff: Learning Generative Models Based on Sinkhorn Divergence

Problem 2: The distance between high-dimensional real and fake data suffers from the curse of dimensionality.

- Train an encoder f for dimensionality reduction in an adversarial way

$$\begin{aligned}&\sup_f \inf_g \bar{W}_{2,\epsilon}(f_{\#}p_x, f_{\#}p_g)\\&= \sup_f \inf_{g,\pi\in\Pi_\epsilon(f_{\#}p_x, f_{\#}p_g)} \mathbb{E}_{f(x),f(g(z))\sim\pi}[\|f(x)-f(g(z))\|_2^2]\end{aligned} \tag{47}$$

- Given a batch of samples,

$$\max_f \min_{g,\boldsymbol{T}\in\Pi_\epsilon} \langle \boldsymbol{D}(f(\boldsymbol{X}), f(g(\boldsymbol{Z}))), \boldsymbol{T}\rangle + \epsilon H(\boldsymbol{T}). \tag{48}$$

- This method can be extended by the OT-GAN in [Salimans et al., 2018], replacing the Sinkhorn divergence to a minibatch energy distance.

[Salimans et al., 2018] Salimans, T., Zhang, H., Radford, A. and Metaxas, D., Improving GANs Using Optimal Transport. ICLR, 2018.

Conditional Transport (CT): Amortization of Approximated OT Plan

Motivation: The OT plan in SinkDiff is nonparametric. Calling Sinkhorn-scaling algorithm is time-consuming.

- **Conditional Transport (CT)** [Zheng et al., 2021]: Relax the OT plan to two coupled transition matrices (with two one-side constraints):

$$\begin{aligned}
&\sup_f \inf_g CT(f_{\#}p_x, f_{\#}p_g) \\
&= \sup_f \inf_{g,\pi_1\in\Pi(p_x,\cdot),\pi_2\in\Pi(\cdot,p_g)} \mathbb{E}_{x,g(z)\sim\pi_1}[d(f(x), f(g(z)))] \\
&\qquad\qquad + \mathbb{E}_{x,g(z)\sim\pi_2}[d(f(x), f(g(z)))].
\end{aligned} \tag{49}$$

[Zheng et al., 2021] Zheng, H. and Zhou, M., Exploiting Chain Rule and Bayes' Theorem to Compare Probability Distributions. NeurIPS, 2021.

Conditional Transport (CT): Amortization of Approximated OT Plan

- Given samples, π_1 and π_2 are parametrized by a softmax-based model (**amortization**):

$$\begin{aligned}\pi_1(x_n|g(z);\phi) &= \frac{\exp\langle\phi(x_n),\phi(g(z))\rangle}{\sum_m \exp\langle\phi(x_m),\phi(g(z))\rangle},\\ \pi_2(g(z_n)|x;\phi) &= \frac{\exp\langle\phi(g(z_n)),\phi(x)\rangle}{\sum_m \exp\langle\phi(g(z_m)),\phi(x)\rangle}.\end{aligned} \tag{50}$$

Conditional Transport (CT): Amortization of Approximated OT Plan

- Given samples, π_1 and π_2 are parametrized by a softmax-based model (**amortization**):

$$\begin{aligned}\pi_1(x_n|g(z);\phi) &= \frac{\exp\langle\phi(x_n),\phi(g(z))\rangle}{\sum_m \exp\langle\phi(x_m),\phi(g(z))\rangle},\\ \pi_2(g(z_n)|x;\phi) &= \frac{\exp\langle\phi(g(z_n)),\phi(x)\rangle}{\sum_m \exp\langle\phi(g(z_m)),\phi(x)\rangle}.\end{aligned} \tag{50}$$

- The learning paradigm is

$$\max_f \min_{g,\phi} \langle \boldsymbol{D}(f(\boldsymbol{X}), f(g(\boldsymbol{Z}))), \boldsymbol{T}_\phi(\boldsymbol{X}, g(\boldsymbol{Z}))\rangle \tag{51}$$

where $\boldsymbol{T}_\phi = [\pi_1(x_n|g(z_m);\phi)]$ or $[\pi_2(g(z_n)|x_m;\phi)]$.

Wasserstein Autoencoder (WAE)

Besides approximate the primal form of W_p by EOT, another way is applying the autoencoding architecture.

Wasserstein Autoencoder (WAE)

Besides approximate the primal form of W_p by EOT, another way is applying the autoencoding architecture.

- **Wasserstein autoencoder (WAE)** [Tolstikhin, et al., 2018] fits the model distribution p_g by minimizing its W_2 distance to the data distribution p_x approximately.

Wasserstein Autoencoder (WAE)

Besides approximate the primal form of W_p by EOT, another way is applying the autoencoding architecture.

- **Wasserstein autoencoder (WAE)** [Tolstikhin, et al., 2018] fits the model distribution p_g by minimizing its W_2 distance to the data distribution p_x approximately.

$$\inf_g W_2(p_x, p_g) \approx \inf_{g,f} \underbrace{\mathbb{E}_{p_x}\mathbb{E}_{q_{z|x;f}}[d_x(x, g(z))]}_{\text{reconstruction loss}} + \underbrace{\gamma d_p(\overbrace{\mathbb{E}_{p_x}[q_{z|x;f}]}^{q_{z;f}}, p_z)}_{\text{distance(posterior, prior)}}, \tag{52}$$

- $q_{z|x;f}$ is the posterior of z given x, parameterized by an **encoder** $f : \mathcal{X} \mapsto \mathcal{Z}$.
- $q_{z;f} = \mathbb{E}_{p_x}[q_{z|x;f}]$ is the expectation of the posterior distributions.
- p_z is the prior of z.

[Tolstikhin, et al., 2018] Tolstikhin, I., Bousquet, O., Gelly, S., & Schoelkopf, B. Wasserstein Auto-Encoders. ICLR 2018.

Comparisons with Other Autoencoders

Method	$f : \mathcal{X} \mapsto \mathcal{Z}$	Prior p_z	Learn p_z	$d_p(q_{z;Q}, p_z)$
VAE	Probabilistic	$\mathcal{N}(z; 0, I)$	No	KL
GMVAE	Probabilistic	$\frac{1}{K}\sum_k \mathcal{N}(z; u_k, \Sigma_k)$	No	KL
VampPrior	Probabilistic	$\frac{1}{K}\sum_k \mathcal{N}(z; Q(x_k))$	Yes	KL

Comparisons with Other Autoencoders

Method	$f : \mathcal{X} \mapsto \mathcal{Z}$	Prior p_z	Learn p_z	$d_p(q_{z;Q}, p_z)$
VAE	Probabilistic	$\mathcal{N}(z; 0, I)$	No	KL
GMVAE	Probabilistic	$\frac{1}{K}\sum_k \mathcal{N}(z; u_k, \Sigma_k)$	No	KL
VampPrior	Probabilistic	$\frac{1}{K}\sum_k \mathcal{N}(z; Q(x_k))$	Yes	KL
WAE	Deterministic	$\mathcal{N}(z; 0, I)$	No	MMD/GAN
SWAE	Deterministic	$\mathcal{N}(z; 0, I)$	No	SW_2
RAE	Probabilistic/Deterministic	$\frac{1}{K}\sum_k \mathcal{N}(z; u_k, \Sigma_k)$	Yes	FGW_2
HCP-AE	Probabilistic/Deterministic	$\mathcal{N}(z; 0, I)$	No	HCP_2

SWAE [Kolouri et al., 2018] Kolouri, S., Pope, P.E., Martin, C.E. and Rohde, G.K., Sliced Wasserstein auto-encoders. ICLR, 2018.

RAE [Xu et al., 2020] Xu, H., Luo, D., Henao, R., Shah, S. and Carin, L., Learning autoencoders with relational regularization. ICML, 2020.

HCP-AE [Li et al., 2022] Li, T., Meng, C., Xu, H. and Yu, J., Hilbert curve projection distance for distribution comparison. arXiv:2205.15059. 2022.

Sliced Wasserstein Generative (SWG) Model

Motivations: Apply a computationally-efficient surrogate of W_p to simply the learning of generative models.

Sliced Wasserstein Generative (SWG) Model

Motivations: Apply a computationally-efficient surrogate of W_p to simply the learning of generative models.

- **Sliced Wasserstein Generative (SWG) Model** [Deshpande et al., 2018]: Minimize SW_2 between p_g and p_x directly as

$$\begin{aligned} \inf_g SW_2(p_x, p_g) &= \inf_g \int_{f \in \mathcal{S}^{D-1}} W_2(f_{\#}p_x, f_{\#}p_g) \mathrm{d}p(f) \\ &= \inf_g \int_{f \in \mathcal{S}^{D-1}} W_2(f_{\#}p_x, f \circ g_{\#}p_z) \mathrm{d}p(f) \end{aligned} \tag{53}$$

Sliced Wasserstein Generative (SWG) Model

Motivations: Apply a computationally-efficient surrogate of W_p to simply the learning of generative models.

- **Sliced Wasserstein Generative (SWG) Model** [Deshpande et al., 2018]: Minimize SW_2 between p_g and p_x directly as

$$\begin{aligned}\inf_g SW_2(p_x, p_g) &= \inf_g \int_{f\in\mathcal{S}^{D-1}} W_2(f_{\#}p_x, f_{\#}p_g)\mathrm{d}p(f) \\ &= \inf_g \int_{f\in\mathcal{S}^{D-1}} W_2(f_{\#}p_x, f\circ g_{\#}p_z)\mathrm{d}p(f)\end{aligned} \tag{53}$$

- Given a batch of samples $\boldsymbol{X}$, we sample L projectors in $\mathcal{S}^{D-1}$ uniformly:

$$\min_g \sum_{l=1}^{L} \widehat{W}_2(f_l(\boldsymbol{X}), f_l(g(\boldsymbol{Z}))), \quad \boldsymbol{Z}\sim\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}). \tag{54}$$

[Deshpande et al., 2018] Deshpande, I., Zhang, Z. and Schwing, A.G., 2018. Generative modeling using the sliced wasserstein distance. CVPR, 2018.

Max-Sliced Wasserstein Generative (Max-SWG) Model

Motivations: MSW_p is strongly equivalent to W_p, so using it as the surrogate can be better.

Max-Sliced Wasserstein Generative (Max-SWG) Model

Motivations: MSW_p is strongly equivalent to W_p, so using it as the surrogate can be better.

- **Max-Sliced Wasserstein Generative (Max-SWG) Model** [Deshpande et al., 2019]: Minimize MSW_2 between p_g and p_x directly as

$$\begin{aligned} \inf_g MSW_2(p_x, p_g) &= \inf_g \sup_{f \in \mathcal{S}^{D-1}} W_2(f_\# p_x, f_\# p_g) \\ &= \inf_g \sup_{f \in \mathcal{S}^{D-1}} W_2(f_\# p_x, f \circ g_\# p_z) \end{aligned} \tag{55}$$

Max-Sliced Wasserstein Generative (Max-SWG) Model

Motivations: MSW_p is strongly equivalent to W_p, so using it as the surrogate can be better.

- **Max-Sliced Wasserstein Generative (Max-SWG) Model** [Deshpande et al., 2019]: Minimize MSW_2 between p_g and p_x directly as

$$\begin{aligned}\inf_g MSW_2(p_x, p_g) &= \inf_g \sup_{f\in\mathcal{S}^{D-1}} W_2(f_\# p_x, f_\# p_g) \\ &= \inf_g \sup_{f\in\mathcal{S}^{D-1}} W_2(f_\# p_x, f\circ g_\# p_z)\end{aligned} \tag{55}$$

- Given a batch of samples $\boldsymbol{X}$, we obtain an adversarial learning paradigm, where f works as the discriminator:

$$\min_g \max_{f\in\mathcal{S}^{D-1}} \widehat{W}_2(f(\boldsymbol{X}), f(g(\boldsymbol{Z}))), \quad \boldsymbol{Z}\sim\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}). \tag{56}$$

[Deshpande et al., 2019] Deshpande, I., Hu, Y.T., Sun, R., Pyrros, A., Siddiqui, N., Koyejo, S., Zhao, Z., Forsyth, D. and Schwing, A.G., Max-sliced wasserstein distance and its use for gans. CVPR, 2019.

Amortized Max-Sliced Wasserstein Generative Model

Motivations: The f in Max-SWG is nonparametric. For each batch, we have to solve an optimization problem iteratively to obtain f.

Amortized Max-Sliced Wasserstein Generative Model

Motivations: The f in Max-SWG is nonparametric. For each batch, we have to solve an optimization problem iteratively to obtain f.

- **Amortized Max-Sliced Wasserstein Generative (AM-SWG) Model** [Nguyen et al., 2022]: Minimize MSW_2 between p_g and p_x **approximately** by applying a parametric projector $f_\theta : \mathcal{X}^{2B} \mapsto \mathcal{S}^{D-1}$, where B is batch size.

$$\begin{aligned}\inf_g \widetilde{MSW}_2(p_x, p_g) &= \inf_g \sup_{f_\theta : \mathcal{X}^{2B} \mapsto \mathcal{S}^{D-1}} W_2(f_{\theta\#}p_x, f_{\theta\#}p_g) \\ &= \inf_g \sup_{f : \mathcal{X}^{2B} \mapsto \mathcal{S}^{D-1}} W_2(f_\#p_x, f_\theta \circ g_\#p_z)\end{aligned} \tag{57}$$

Amortized Max-Sliced Wasserstein Generative Model

Motivations: The f in Max-SWG is nonparametric. For each batch, we have to solve an optimization problem iteratively to obtain f.

- **Amortized Max-Sliced Wasserstein Generative (AM-SWG) Model** [Nguyen et al., 2022]: Minimize MSW_2 between p_g and p_x **approximately** by applying a parametric projector $f_\theta : \mathcal{X}^{2B} \mapsto \mathcal{S}^{D-1}$, where B is batch size.

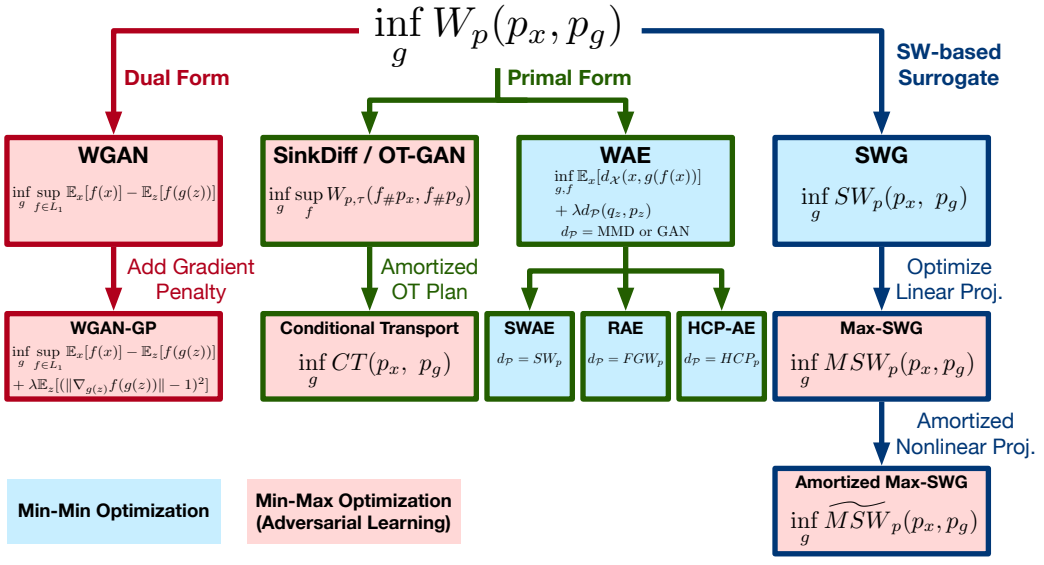
$$\begin{aligned}\inf_g \widetilde{MSW}_2(p_x, p_g) &= \inf_g \sup_{f_\theta : \mathcal{X}^{2B} \mapsto \mathcal{S}^{D-1}} W_2(f_{\theta\#}p_x, f_{\theta\#}p_g) \\ &= \inf_g \sup_{f : \mathcal{X}^{2B} \mapsto \mathcal{S}^{D-1}} W_2(f_{\#}p_x, f_\theta \circ g_{\#}p_z)\end{aligned} \tag{57}$$

- Given a batch of samples $\boldsymbol{X}$, we obtain an adversarial learning paradigm, where f works as the discriminator:

$$\min_g \max_\theta \widehat{W}_2(f_\theta(\boldsymbol{X}), f_\theta(g(\boldsymbol{Z}))), \quad \boldsymbol{Z} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}). \tag{58}$$

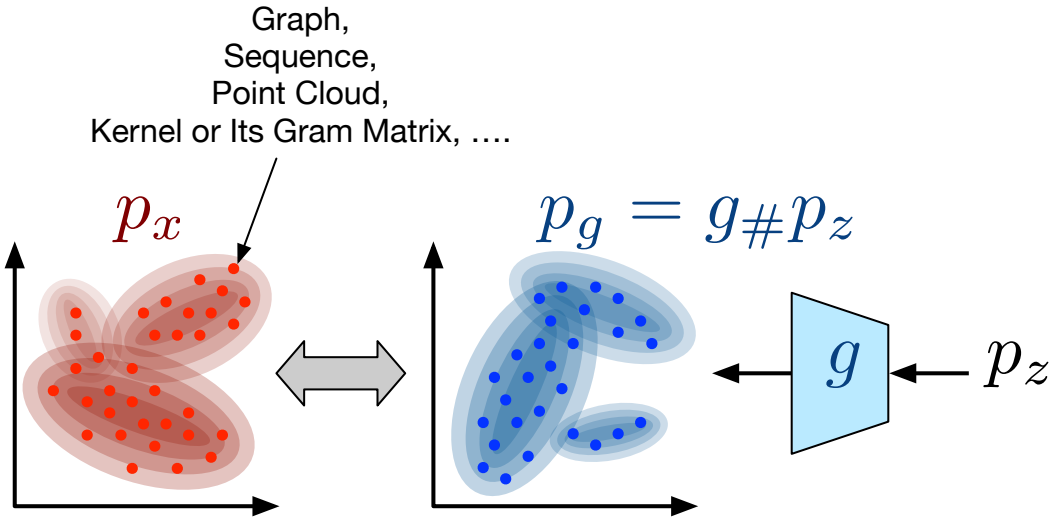
[Nguyen et al., 2022] Nguyen, K. and Ho, N., Amortized projection optimization for sliced Wasserstein generative models. NeurIPS, 2022.

A (Partial) Family Tree of OT-based Generative Models

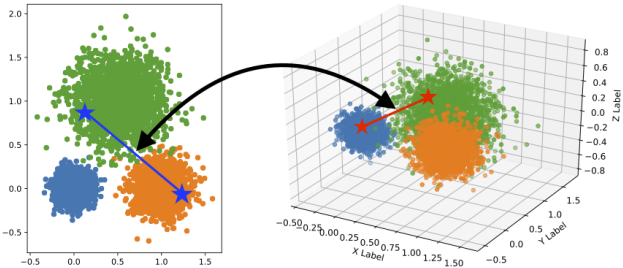


Long break

Recent Progress: OT-based Generative Models for Structured Data



Gromov-Wasserstein Distance: The OT between MM-Spaces



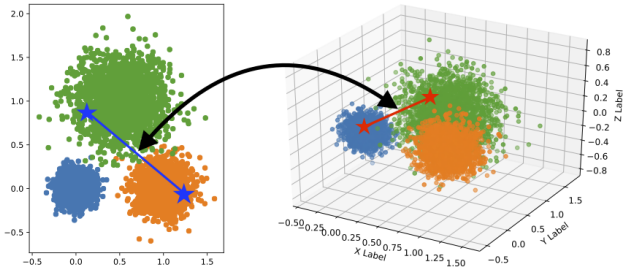
Gromov-Wasserstein distance [Sturm 2006; Mémoli 2011] between $\mathcal{X}_{d_X,\mu_X}$ and $\mathcal{Y}_{d_Y,\mu_Y}$:

$$GW_p(\mathcal{X}_{d_X,\mu_X},\mathcal{Y}_{d_Y,\mu_Y}) := \left(\inf_{\pi\in\Pi(\mu_X,\mu_Y)} \mathbb{E}_{x,x',y,y'\sim\pi\times\pi}[r(x,x',y,y')]\right)^{1/p}$$
$$= \left(\inf_{\pi\in\Pi(\mu_X,\mu_Y)} \int_{(x,x')\in\mathcal{X}^2}\int_{(y,y')\in\mathcal{Y}^2} r(x,x',y,y')\mathrm{d}\pi(x,y)\mathrm{d}\pi(x',y')\right)^{1/p}. \quad (59)$$

Minimize expected relational distance $r(x,x',y,y') = |d_X(x,x') - d_Y(y,y')|^p$.

[Sturm 2006] Sturm, K.T., On the geometry of metric measure spaces. Acta Mathematica, 2006.
[Mémoli 2011] Mémoli, F., Gromov–Wasserstein distances and the metric approach to object matching. Foundations of computational mathematics, 2011.

Gromov-Wasserstein Distance for Structured Data (Point Clouds)

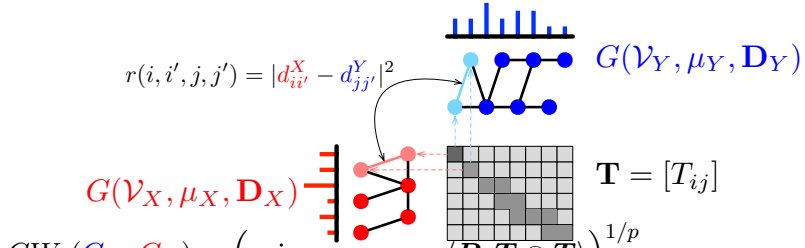


Given $\boldsymbol{X} = \{x_m\}_{m=1}^{M}$, $\boldsymbol{Y} = \{y_n\}_{n=1}^{N}$, and distributions $\boldsymbol{\mu}_X \in \Delta^{N-1}$, $\boldsymbol{\mu}_Y \in \Delta^{M-1}$:

$$\begin{aligned}\widehat{GW}_p(\boldsymbol{X}, \boldsymbol{Y}) &= \left(\min_{\boldsymbol{T} \in \Pi(\boldsymbol{\mu}_X, \boldsymbol{\mu}_Y)} \langle \boldsymbol{R}, \boldsymbol{T} \otimes \boldsymbol{T} \rangle\right)^{1/p} \\ &= \left(\min_{\boldsymbol{T} \in \Pi(\boldsymbol{\mu}_X, \boldsymbol{\mu}_Y)} \sum_{m,m'=1}^{M} \sum_{n,n'=1}^{N} r(x_m, x_{m'}, y_n, y_{n'}) t_{mn} t_{m'n'}\right)^{1/p}.\end{aligned} \tag{60}$$

- π^* or $\boldsymbol{T}^*$: the optimal transport plan between samples.
- $\pi^* \times \pi^*$ or $\boldsymbol{T}^* \otimes \boldsymbol{T}^*$: the optimal transport plan between sample pairs.
- **Useful properties: Translation-, rotation-, and permutation-invariance**

Gromov-Wasserstein Distance for Structured Data (Graphs)

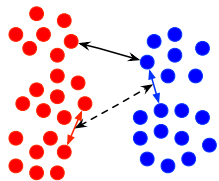


- **A (pseudo) metric for graphs: permutation-invariant, robust to graph size, more efficient than QAP.**

[Chowdhury et al., 2019] Chowdhury, S., & Mémoli, F. The gromov–wasserstein distance between networks and stable network invariants. Information and Inference: A Journal of the IMA, 2019.
[Xu et al., 2019] Xu, H., Luo, D., & Carin, L. Gromov-Wasserstein learning for graph matching and node embedding. ICML, 2019.

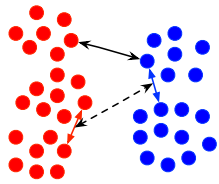
Fused Gromov-Wasserstein Distance: Combine GW_p with W_p

For graphs/point clouds, consider the GW distance between their topological structure/nodes' relations and the Wasserstein distance between their node attributions jointly [Titouan, et al., 2019].



Fused Gromov-Wasserstein Distance: Combine GW_p with W_p

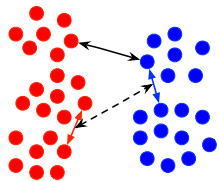
For graphs/point clouds, consider the GW distance between their topological structure/nodes' relations and the Wasserstein distance between their node attributions jointly [Titouan, et al., 2019].



$$FGW_p(\mathcal{X}_{d_X,\mu_X}, \mathcal{Y}_{d_Y,\mu_Y}; \beta) = \Big(\inf_{\pi \in \Pi(\mu_X,\mu_Y)} (1-\beta) \underbrace{\int_{\mathcal{X}\times\mathcal{Y}} d(x,y) \mathrm{d}\pi(x,y)}_{\text{Wasserstein term}} + \beta \underbrace{\int_{\mathcal{X}\times\mathcal{Y}} \int_{\mathcal{X}\times\mathcal{Y}} r_{x,y,x',y'} \mathrm{d}\pi(x,y) \mathrm{d}\pi(x',y')}_{\text{Gromov-Wasserstein term}} \Big)^{1/p}. \tag{62}$$

[Titouan, et al., 2019] Titouan, V., Courty, N., Tavenard, R., & Flamary, R. Optimal transport for structured data with application on graphs. ICML 2019.

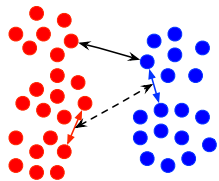
Fused Gromov-Wasserstein Distance: Combine GW_p with W_p



In practice, given $\boldsymbol{X} = \{x_i\}_{i=1}^M$ and $\boldsymbol{Y} = \{y_j\}_{j=1}^N$, we have

$$\widehat{FGW}_p(\boldsymbol{X}, \boldsymbol{Y}; \beta) = \Big(\min_{\boldsymbol{T} \in \Pi(\boldsymbol{\mu}_X, \boldsymbol{\mu}_Y)} (1-\beta) \sum_{m=1}^M \sum_{n=1}^N d(x_m, y_n) t_{mn} + \beta \sum_{m,m'=1}^M \sum_{n,n'=1}^N r(x_m, x_{m'}, y_n, y_{n'}) t_{mn} t_{m'n'} \Big)^{1/p}$$

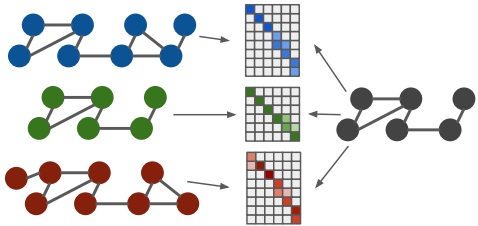
Fused Gromov-Wasserstein Distance: Combine GW_p with W_p



In practice, given $\boldsymbol{X} = \{x_i\}_{i=1}^M$ and $\boldsymbol{Y} = \{y_j\}_{j=1}^N$, we have

$$\begin{aligned}\widehat{FGW}_p(\boldsymbol{X}, \boldsymbol{Y}; \beta) =& \Big(\min_{\boldsymbol{T} \in \Pi(\boldsymbol{\mu}_X, \boldsymbol{\mu}_Y)} (1-\beta) \sum_{m=1}^M \sum_{n=1}^N d(x_m, y_n) t_{mn} \\ &+ \beta \sum_{m,m'=1}^M \sum_{n,n'=1}^N r(x_m, x_{m'}, y_n, y_{n'}) t_{mn} t_{m'n'} \Big)^{1/p} \\ \Leftrightarrow& \big(\min_{\boldsymbol{T} \in \Pi(\boldsymbol{\mu}_X, \boldsymbol{\mu}_Y)} \langle \boldsymbol{D}_{XY} - 2\boldsymbol{D}_X \boldsymbol{T} \boldsymbol{D}_Y^\top, \boldsymbol{T} \rangle \big)^{1/p}\end{aligned} \tag{63}$$

From (F)GW Distance to (F)GW Barycenters

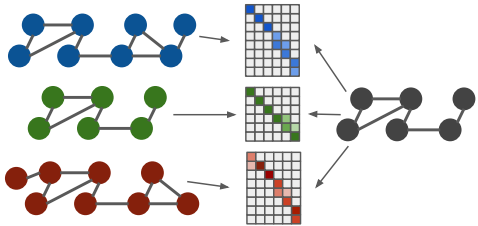


Given $\{G_k\}_{k=1}^K$, $K \geq 2$, their (weighted) GW barycenter [Peyré, et al., 2016] is

$$\underbrace{\bar{G}(\bar{\mathcal{V}}, \bar{\boldsymbol{\mu}}, \boldsymbol{B}^*)}_{\text{Barycenter graph}}, \ \underbrace{\{\boldsymbol{T}_k^*\}_{k=1}^K}_{\text{OT matrices}} := \arg\min_G \sum_{k=1}^K \lambda_k GW_p^p(G_k, G) \tag{64}$$

$$\Leftrightarrow \arg\min_{\boldsymbol{B}} \min_{\{\boldsymbol{T}_k \in \Pi(\boldsymbol{\mu}_k, \bar{\boldsymbol{\mu}})\}_{k=1}^K} -\sum_{k=1}^K \lambda_k \langle \boldsymbol{D}_k \boldsymbol{T}_k \boldsymbol{B}^\top, \boldsymbol{T}_k \rangle$$

From (F)GW Distance to (F)GW Barycenters



Given $\{G_k\}_{k=1}^K$, $K \geq 2$, their (weighted) GW barycenter [Peyré, et al., 2016] is

$$\underbrace{\bar{G}(\bar{\mathcal{V}}, \bar{\boldsymbol{\mu}}, \boldsymbol{B}^*)}_{\text{Barycenter graph}}, \quad \underbrace{\{\boldsymbol{T}_k^*\}_{k=1}^K}_{\text{OT matrices}} := \arg\min_G \sum_{k=1}^K \lambda_k GW_p^p(G_k, G) \tag{64}$$

$$\Leftrightarrow \arg\min_{\boldsymbol{B}} \min_{\{\boldsymbol{T}_k \in \Pi(\boldsymbol{\mu}_k, \bar{\boldsymbol{\mu}})\}_{k=1}^K} -\sum_{k=1}^K \lambda_k \langle \boldsymbol{D}_k \boldsymbol{T}_k \boldsymbol{B}^\top, \boldsymbol{T}_k \rangle$$

Permutation-invariance: If $\bar{G}$ is a GW barycenter of $\{G_k\}_{k=1}^K$, then permute($\bar{G}$) is a valid GW barycenter as well.

Implementation of GW Barycenter

- Approximation of $\bar{\boldsymbol{\mu}}$ [Xu, et al., 2019]:

$$\bar{\boldsymbol{\mu}} = \sum_{k=1}^{K} \lambda_k \text{interpolate}_{|\bar{\mathcal{V}}|}(\text{sort}(\boldsymbol{\mu}_k)), \tag{65}$$

  - $\text{sort}(\cdot)$ sorts the elements of the input vector in descending order.
  - $\text{interpolate}_{|\bar{\mathcal{V}}|}(\cdot)$ samples $|\bar{\mathcal{V}}|$ values from the input vector via an interpolation method (e.g., bilinear or cubic interpolation).

Implementation of GW Barycenter

- Approximation of $\bar{\boldsymbol{\mu}}$ [Xu, et al., 2019]:

$$\bar{\boldsymbol{\mu}} = \sum\nolimits_{k=1}^{K} \lambda_k \text{interpolate}_{|\bar{\mathcal{V}}|}(\text{sort}(\boldsymbol{\mu}_k)), \tag{65}$$

  - $\text{sort}(\cdot)$ sorts the elements of the input vector in descending order.
  - $\text{interpolate}_{|\bar{\mathcal{V}}|}(\cdot)$ samples $|\bar{\mathcal{V}}|$ values from the input vector via an interpolation method (e.g., bilinear or cubic interpolation).

- **Alternating optimization strategy:**
  - Obtain $\boldsymbol{T}_k = \arg\min_{\boldsymbol{T} \in \Pi(\boldsymbol{\mu}_k, \bar{\boldsymbol{\mu}})} -\langle \boldsymbol{D}_k \boldsymbol{T} \boldsymbol{B}^\top, \boldsymbol{T} \rangle$ for $k = 1, ..., K$.
    - Sinkhorn, Proximal Gradient, Bregman ADMM, ...
  - Update barycenter in a closed form (the first-order optimality condition):

$$\boldsymbol{B}^* = \frac{1}{\bar{\boldsymbol{\mu}} \bar{\boldsymbol{\mu}}^\top} \sum\nolimits_{k=1}^{K} \lambda_k \boldsymbol{T}_k^\top \boldsymbol{D}_k \boldsymbol{T}_k. \tag{66}$$

[Xu, et al. 2019] Xu, H., Luo, D., & Carin, L. Scalable Gromov-Wasserstein learning for graph partitioning and matching. NeurIPS, 2019.

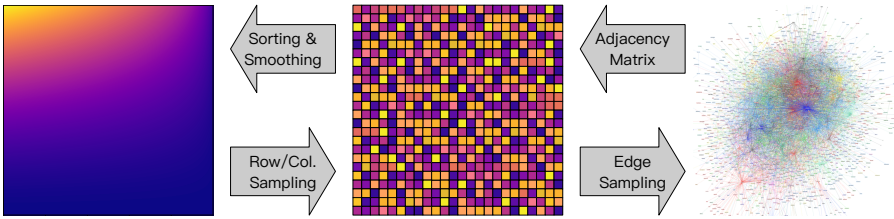
Open Resources

AAAI'22 Tutorial on Gromov-Wasserstein Learning
https://hongtengxu.github.io/talks.html

OT-based Graph Generative Modeling

Leverage (F)GW distance and barycenters, we can

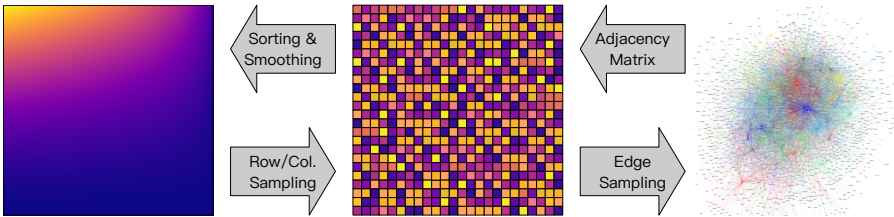
- **Estimate Graphons (Nonparametric Graph Generative Models)**



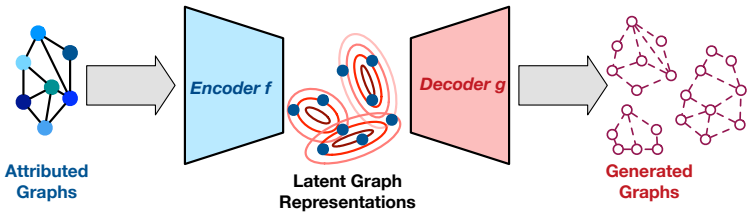
OT-based Graph Generative Modeling

Leverage (F)GW distance and barycenters, we can

- **Estimate Graphons (Nonparametric Graph Generative Models)**

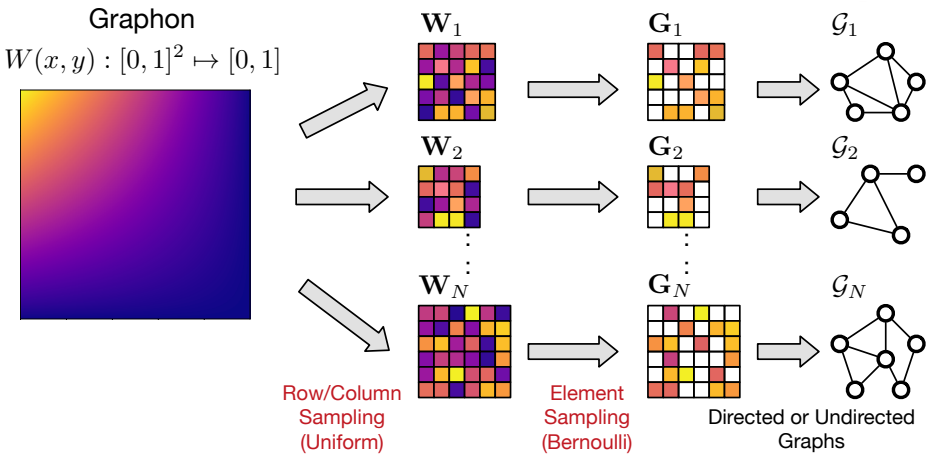


- **Learn Graph-level Wasserstein Autoencoders**



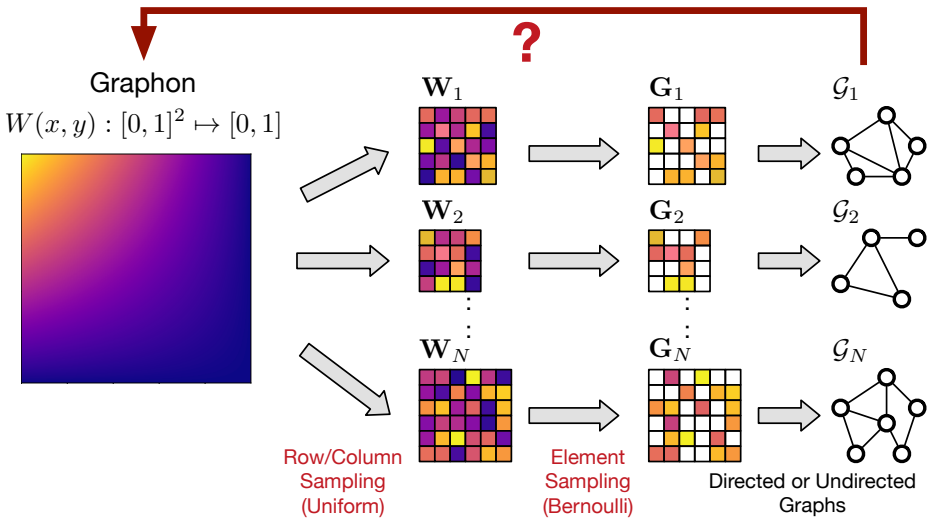
Graphon Estimation: Learning A Nonparametric Graph Generator

Graphon: A Nonparametric Graph Generative Model



Graphon Estimation: Learning A Nonparametric Graph Generator

Graphon: A Nonparametric Graph Generative Model



Two Paradigms of Graphon Estimation

- **Traditional paradigm:** Given a single large-scale graph, estimate a graphon by a step-function and make it as smooth as possible.
  - Collecting and processing a large-scale graph are challenging.
  - The estimation and its smoothness depend on the sorting of nodes (according to their degrees).

Two Paradigms of Graphon Estimation

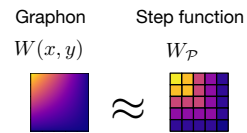
- **Traditional paradigm:** Given a single large-scale graph, estimate a graphon by a step-function and make it as smooth as possible.
  - Collecting and processing a large-scale graph are challenging.
  - The estimation and its smoothness depend on the sorting of nodes (according to their degrees).
- **The proposed paradigm:** Given a set of unaligned but small graphs, estimate a graphon by solving a GW barycenter problem [Xu, et al., 2021].
  - Reduce the difficulty on data collection and processing.
  - Robust to the challenging cases where the graph nodes are hard to sort.

[Xu, et al., 2021] Xu, H., Luo, D., Carin, L., & Zha, H. Learning Graphons via Structured Gromov-Wasserstein Barycenters. AAAI 2021.

Oracle Graphon Estimation

$$\textbf{Cut norm: } \|W\|_{\square} := \sup_{\mathcal{X},\mathcal{Y}\subset\Omega} \Big| \int_{\mathcal{X}\times\mathcal{Y}} W(x,y)\mathrm{d}x\mathrm{d}y \Big|,$$

$$\textbf{Cut distance: } \delta_{\square}(W_1,W_2) := \inf_{\underbrace{\phi \in \mathcal{S}_\Omega}_{\text{measure-preserved map}}} \| \underbrace{W_1 - W_2^{\phi}}_{\text{Residual}} \|_{\square}. \tag{67}$$



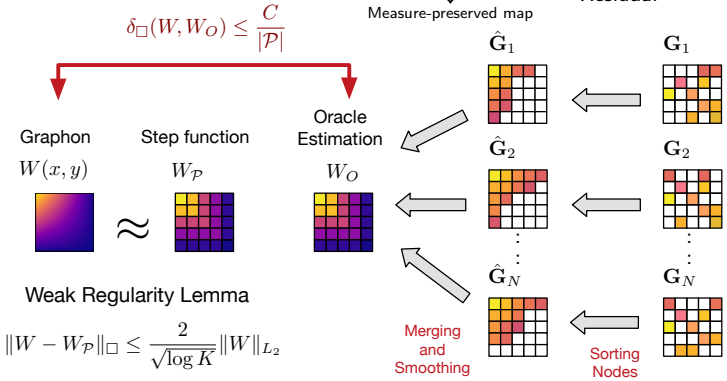
Weak Regularity Lemma

$$\|W - W_{\mathcal{P}}\|_{\square} \leq \frac{2}{\sqrt{\log K}} \|W\|_{L_2}$$

Oracle Graphon Estimation

Cut norm: $\|W\|_\square := \sup_{\mathcal{X},\mathcal{Y}\subset\Omega}\left|\int_{\mathcal{X}\times\mathcal{Y}} W(x,y)\mathrm{d}x\mathrm{d}y\right|,$

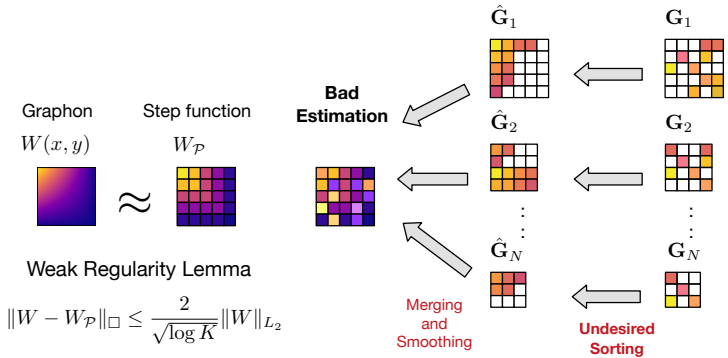
Cut distance: $\delta_\square(W_1,W_2) := \underbrace{\inf_{\phi\in\mathcal{S}_\Omega}}_{\text{Measure-preserved map}} \underbrace{\|W_1 - W_2^\phi\|_\square}_{\text{Residual}}.$ (67)



Oracle Graphon Estimation

Cut norm: $\|W\|_{\square} := \sup_{\mathcal{X},\mathcal{Y}\subset\Omega}\left|\int_{\mathcal{X}\times\mathcal{Y}} W(x,y)\mathrm{d}x\mathrm{d}y\right|,$

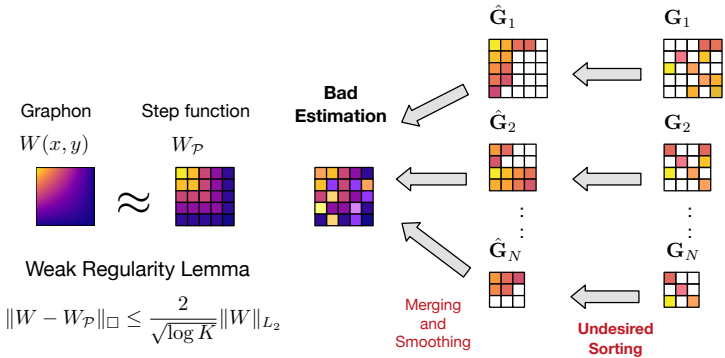
Cut distance: $\delta_{\square}(W_1,W_2) := \inf_{\underbrace{\phi\in\mathcal{S}_\Omega}_{\text{leasure-preserved map}}} \|\underbrace{W_1 - W_2^{\phi}}_{\text{Residual}}\|_{\square}.$ (67)



Oracle Graphon Estimation

Cut norm: $\|W\|_\square := \sup_{\mathcal{X},\mathcal{Y}\subset\Omega}\left|\int_{\mathcal{X}\times\mathcal{Y}} W(x,y)\mathrm{d}x\mathrm{d}y\right|,$

Cut distance: $\delta_\square(W_1,W_2) := \inf_{\underbrace{\phi\in\mathcal{S}_\Omega}_{\text{asure-preserved map}}} \|\underbrace{W_1 - W_2^\phi}_{\text{Residual}}\|_\square.$ (67)



Can we achieve graph alignment and graphon learning jointly?

Practical Estimation: Optimizing An Upper Bound of Estimation Error

Learning task

- $W : \Omega^2 \mapsto [0, 1]$: An unknown target graphon

Data

- $\{G_m : \Omega^2 \mapsto [0, 1]\}_{m=1}^{M}$: The step functions of observed graphs (induced from adjacency matrices)

Practical Estimation: Optimizing An Upper Bound of Estimation Error

Learning task

- $W : \Omega^2 \mapsto [0, 1]$: An unknown target graphon

Data

- $\{G_m : \Omega^2 \mapsto [0, 1]\}_{m=1}^M$: The step functions of observed graphs (induced from adjacency matrices)

Unknown but useful concepts for theoretical analysis

- $\{\widehat{G}_m : \Omega^2 \mapsto [0, 1]\}_{m=1}^M$: Perfectly aligned step functions of observed graphs (by perfect measure-preserving mapping)
- $W_O = \frac{1}{M} \sum_{m=1}^M \widehat{G}_m$: The oracle graph estimator

Practical Estimation: Optimizing An Upper Bound of Estimation Error

Learning task

- $W : \Omega^2 \mapsto [0, 1]$: An unknown target graphon

Data

- $\{G_m : \Omega^2 \mapsto [0, 1]\}_{m=1}^M$: The step functions of observed graphs (induced from adjacency matrices)

Unknown but useful concepts for theoretical analysis

- $\{\widehat{G}_m : \Omega^2 \mapsto [0, 1]\}_{m=1}^M$: Perfectly aligned step functions of observed graphs (by perfect measure-preserving mapping)
- $W_O = \frac{1}{M} \sum_{m=1}^M \widehat{G}_m$: The oracle graph estimator

Practical Implementation

- W_P: The practical estimation based on observed $\{G_m\}_{m=1}^M$.

Practical Estimation: Optimizing An Upper Bound of Estimation Error

$$\begin{aligned}\delta_{\square}(W, W_{\mathcal{P}}) \leq & \delta_{\square}(W, W_O) + \delta_{\square}(W_O, W_{\mathcal{P}}) \qquad \text{(Triangle Inequality)} \\ = & \delta_{\square}(W, W_O) + \delta_{\square}\Big(\frac{1}{M}\sum\nolimits_{m=1}^{M} \hat{G}_m, W_{\mathcal{P}}\Big)\end{aligned}$$

Practical Estimation: Optimizing An Upper Bound of Estimation Error

$$
\begin{aligned}
\delta_{\square}(W, W_{\mathcal{P}}) \leq & \delta_{\square}(W, W_O) + \delta_{\square}(W_O, W_{\mathcal{P}}) && \text{(Triangle Inequality)} \\
= & \delta_{\square}(W, W_O) + \delta_{\square}\Big(\frac{1}{M}\sum\nolimits_{m=1}^{M} \hat{G}_m, W_{\mathcal{P}}\Big) \\
\leq & \delta_{\square}(W, W_O) + \frac{1}{M}\sum\nolimits_{m=1}^{M} \delta_{\square}(\hat{G}_m, W_{\mathcal{P}}) && \text{(Triangle Inequality)}
\end{aligned}
$$

Practical Estimation: Optimizing An Upper Bound of Estimation Error

$$
\begin{aligned}
\delta_{\square}(W, W_{\mathcal{P}}) \leq & \delta_{\square}(W, W_O) + \delta_{\square}(W_O, W_{\mathcal{P}}) && \text{(Triangle Inequality)} \\
= & \delta_{\square}(W, W_O) + \delta_{\square}\Big(\frac{1}{M}\sum\nolimits_{m=1}^{M} \hat{G}_m, W_{\mathcal{P}}\Big) \\
\leq & \delta_{\square}(W, W_O) + \frac{1}{M}\sum\nolimits_{m=1}^{M} \delta_{\square}(\hat{G}_m, W_{\mathcal{P}}) && \text{(Triangle Inequality)} \\
= & \delta_{\square}(W, W_O) + \frac{1}{M}\sum\nolimits_{m=1}^{M} \delta_{\square}(G_m, W_{\mathcal{P}}) && (\delta_{\square}(G_m, \hat{G}_m) = 0)
\end{aligned}
$$

Practical Estimation: Optimizing An Upper Bound of Estimation Error

$$
\begin{aligned}
\delta_{\square}(W, W_{\mathcal{P}}) \leq & \delta_{\square}(W, W_O) + \delta_{\square}(W_O, W_{\mathcal{P}}) && \text{(Triangle Inequality)} \\
= & \delta_{\square}(W, W_O) + \delta_{\square}\Big(\frac{1}{M}\sum\nolimits_{m=1}^{M} \hat{G}_m, W_{\mathcal{P}}\Big) \\
\leq & \delta_{\square}(W, W_O) + \frac{1}{M}\sum\nolimits_{m=1}^{M} \delta_{\square}(\hat{G}_m, W_{\mathcal{P}}) && \text{(Triangle Inequality)} \\
= & \delta_{\square}(W, W_O) + \frac{1}{M}\sum\nolimits_{m=1}^{M} \delta_{\square}(G_m, W_{\mathcal{P}}) && (\delta_{\square}(G_m, \hat{G}_m) = 0) \\
\leq & \delta_{\square}(W, W_O) + \frac{1}{M}\sum\nolimits_{m=1}^{M} \delta_1(G_m, W_{\mathcal{P}}). && (\delta_{\square}(W, W') \leq \delta_1(W, W'))
\end{aligned}
\tag{68}
$$

- δ_1 **distance:** $\delta_1(W_1, W_2) := \inf_{\phi \in \mathcal{S}_\Omega} \|W_1 - W_2^{\phi}\|_{L_1}$.
- **Task:** $\min_{W_{\mathcal{P}} \in [0,1]^{K \times K}} \sum_{m=1}^{M} \delta_1(G_m, W_{\mathcal{P}})$.

Learning Graphons as Gromov-Wasserstein Barycenters

$$\underbrace{\delta_1(W_1, W_2) = GW_1(W_1, W_2)}_{\text{[Janson, 2013]}}$$

Learning Graphons as Gromov-Wasserstein Barycenters

$$\underbrace{\delta_1(W_1, W_2) = GW_1(W_1, W_2)}_{\text{[Janson, 2013]}} \underbrace{\cong GW_2(W_1, W_2)}_{\text{[Mémoli, 2011]}}. \tag{69}$$

Learning Graphons as Gromov-Wasserstein Barycenters

$$\underbrace{\delta_1(W_1, W_2) = GW_1(W_1, W_2)}_{\text{[Janson, 2013]}} \underbrace{\cong GW_2(W_1, W_2)}_{\text{[Mémoli, 2011]}}. \tag{69}$$

The task becomes a **Gromov-Wasserstein barycenter** (**GWB**) problem:

$$\min_{W_{\mathcal{P}} \in [0,1]^{K \times K}} \sum_{m=1}^{M} GW_2^2(G_m, W_{\mathcal{P}}). \tag{70}$$

Learning Graphons as Gromov-Wasserstein Barycenters

$$\underbrace{\delta_1(W_1, W_2) = GW_1(W_1, W_2)}_{\text{[Janson, 2013]}} \underbrace{\cong GW_2(W_1, W_2)}_{\text{[Mémoli, 2011]}}. \tag{69}$$

The task becomes a **Gromov-Wasserstein barycenter** (**GWB**) problem:

$$\min_{W_{\mathcal{P}} \in [0,1]^{K \times K}} \sum_{m=1}^{M} GW_2^2(G_m, W_{\mathcal{P}}). \tag{70}$$

Let $\boldsymbol{A}$ and $\boldsymbol{\mu}$ be the matrix and marginal vectors corresponding to a 2D step function, we have

$$GW_2^2(G_m, W_{\mathcal{P}}) = \min_{\boldsymbol{T} \in \Pi(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2)} \underbrace{\sum_{i,i',j,j'} r_{ii'jj'} \overbrace{t_{ij} t_{i'j'}}^{p(r)}}_{\mathbb{E}[r]} \Leftrightarrow \min_{\boldsymbol{T} \in \Pi(\boldsymbol{\mu}_1, \boldsymbol{\mu}_2)} -\langle \boldsymbol{A}_1 \boldsymbol{T} \boldsymbol{A}_2^\top, \boldsymbol{T} \rangle. \tag{71}$$

[Janson, 2013] Janson, S. Graphons, cut norm and distance, couplings and rearrangements. New York journal of mathematics, 2013.
[Mémoli, 2011] Mémoli, F. Gromov–Wasserstein distances and the metric approach to object matching. Foundations of computational mathematics, 2011.

Two Variants for Structured GWBs

Can we achieve a smoothed graphon?

Two Variants for Structured GWBs

Can we achieve a smoothed graphon?

- **Smoothed GW Barycenter (SGWB)** [Xu, et al., AAAI 2021]:

$$\min_{W_{\mathcal{P}} \in [0,1]^{K \times K}} \sum_{m=1}^{M} GW_2^2(G_m, W_{\mathcal{P}}) + \alpha \|\Delta W_{\mathcal{P}}\|_F^2. \tag{72}$$

Two Variants for Structured GWBs

Can we achieve a smoothed graphon?

- **Smoothed GW Barycenter (SGWB)** [Xu, et al., AAAI 2021]:

$$\min_{W_{\mathcal{P}} \in [0,1]^{K \times K}} \sum_{m=1}^{M} GW_2^2(G_m, W_{\mathcal{P}}) + \alpha \|\Delta W_{\mathcal{P}}\|_F^2. \tag{72}$$

- Similar to classic GWB problem, an alternating optimization strategy works well.

Two Variants for Structured GWBs

Can we achieve a smoothed graphon?

- **Smoothed GW Barycenter (SGWB)** [Xu, et al., AAAI 2021]:

$$\min_{W_{\mathcal{P}} \in [0,1]^{K \times K}} \sum_{m=1}^{M} GW_2^2(G_m, W_{\mathcal{P}}) + \alpha \|\Delta W_{\mathcal{P}}\|_F^2. \tag{72}$$

- Similar to classic GWB problem, an alternating optimization strategy works well.

When the graphs are from multiple graphons, how to learn the model?

Two Variants for Structured GWBs

Can we achieve a smoothed graphon?

- **Smoothed GW Barycenter (SGWB)** [Xu, et al., AAAI 2021]:

$$\min_{W_{\mathcal{P}} \in [0,1]^{K \times K}} \sum_{m=1}^{M} GW_2^2(G_m, W_{\mathcal{P}}) + \alpha \|\Delta W_{\mathcal{P}}\|_F^2. \tag{72}$$

- Similar to classic GWB problem, an alternating optimization strategy works well.

When the graphs are from multiple graphons, how to learn the model?

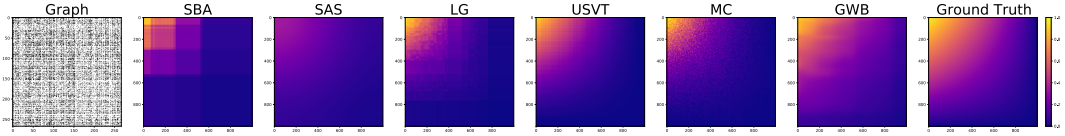
- **A Mixture of GWBs** [Xu, et al., AAAI 2021]:

$$\min_{\{W_c\}_{c=1}^{C}, \boldsymbol{P} \in \Pi(\frac{1}{C}\mathbf{1}_C, \frac{1}{M}\mathbf{1}_M)} \underbrace{\sum_{c=1}^{C} \sum_{m=1}^{M} p_{cm} GW_2^2(G_m, W_c)}_{\langle \boldsymbol{P},\ \boldsymbol{D}_{\mathrm{gw}} \rangle} \tag{73}$$

- p_{cm}: the probability of generating the m-th graph from the c-th graphon.
- Learn a graphon set to minimize its hierarchical GW distance to the observed graphs.

Experiments

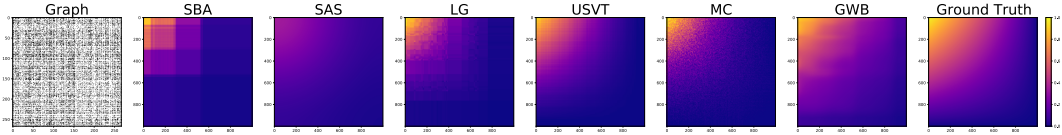
Easy Case: The node degrees provide strong evidence for sorting nodes.



(a) $W(x, y) = xy$ and $0 \leq x, y \leq 1$

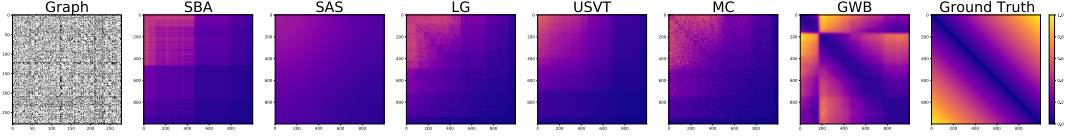
Experiments

Easy Case: The node degrees provide strong evidence for sorting nodes.



(a) $W(x, y) = xy$ and $0 \leq x, y \leq 1$

Hard Case: The nodes of each graph have comparable degrees.



(b) $W(x, y) = |x - y|$ and $0 \leq x, y \leq 1$

Graphon Wasserstein Autoencoder (GWAE) for Graph Generation

Motivations:

- Build a graph-level WAE for graph representation and generation
- Achieve Transferable graph generation (e.g., generating arbitrary-sized graphs with similar node clustering structures)

Graphon Wasserstein Autoencoder (GWAE) for Graph Generation

Motivations:

- Build a graph-level WAE for graph representation and generation
- Achieve Transferable graph generation (e.g., generating arbitrary-sized graphs with similar node clustering structures)

Recall the Wasserstein Autoencoder:

$$\inf_{g} W_2(p_x, p_g) \approx \inf_{g,f} \underbrace{\mathbb{E}_{p_x}\mathbb{E}_{q_{z|x;f}}[d_x(x, g(z))]}_{\text{reconstruction loss}} + \underbrace{\gamma d_p(\overbrace{\mathbb{E}_{p_x}[q_{z|x;f}]}^{q_{z;f}}, p_z)}_{\text{distance(posterior, prior)}}, \tag{74}$$

Reconstruction Loss: x and $g(z)$ becomes graphs, so d_x cannot be Euclidean anymore.

- A nature choose is GW distance.

Graphon Wasserstein Autoencoder (GWAE) for Graph Generation

Motivations:

- Build a graph-level WAE for graph representation and generation
- Achieve Transferable graph generation (e.g., generating arbitrary-sized graphs with similar node clustering structures)

Recall the Wasserstein Autoencoder:

$$\inf_{g} W_2(p_x, p_g) \approx \inf_{g,f} \underbrace{\mathbb{E}_{p_x}\mathbb{E}_{q_{z|x;f}}[d_x(x, g(z))]}_{\text{reconstruction loss}} + \underbrace{\gamma d_p(\overbrace{\mathbb{E}_{p_x}[q_{z|x;f}]}^{q_{z;f}}, p_z)}_{\text{distance(posterior, prior)}}, \tag{74}$$

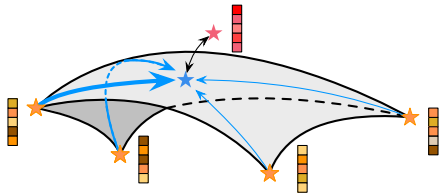
Reconstruction Loss: x and $g(z)$ becomes graphs, so d_x cannot be Euclidean anymore.

- A nature choose is GW distance.

Model Architecture:

- Encoder f can be a GNN
- Decoder g can be a Gromov-Wasserstein Factorization (GWF) model

Gromov-Wasserstein Factorization

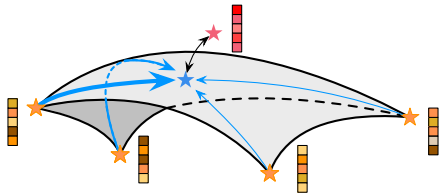


Traditional Factorization Models:

- Given $\{\boldsymbol{y}_1, ..., \boldsymbol{y}_I\}$, we would like to learn the basis $\boldsymbol{A} = [\boldsymbol{a}_1, ..., \boldsymbol{a}_K]$ and the coefficient vector (representation) $\boldsymbol{\lambda}_i$ for each $\boldsymbol{y}_i$.

$$\min_{\{\boldsymbol{A}, \boldsymbol{\lambda}_{1:I}\} \in \Omega} \sum_{i=1}^{I} d_{\text{loss}}(\boldsymbol{A}\boldsymbol{\lambda}_i, \boldsymbol{y}_i). \tag{75}$$

Gromov-Wasserstein Factorization



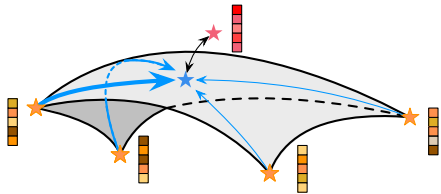
Traditional Factorization Models:

- Given $\{\boldsymbol{y}_1, ..., \boldsymbol{y}_I\}$, we would like to learn the basis $\boldsymbol{A} = [\boldsymbol{a}_1, ..., \boldsymbol{a}_K]$ and the coefficient vector (representation) $\boldsymbol{\lambda}_i$ for each $\boldsymbol{y}_i$.

$$\min_{\{\boldsymbol{A}, \boldsymbol{\lambda}_{1:I}\} \in \Omega} \sum_{i=1}^{I} d_{\text{loss}}(\boldsymbol{A}\boldsymbol{\lambda}_i, \boldsymbol{y}_i). \tag{75}$$

  - PCA: $d_{\text{loss}} = \ell_2$. (MSE)
  - Robust PCA: $d_{\text{loss}} = \ell_1$. (MAE)
  - NMF: $d_{\text{loss}} = \ell_2$, $\Omega =$ Nonnegativeness.
  - LDA: $d_{\text{loss}} =$ KL, $\Omega =$ Simplex.
  - Wasserstein dictionary learning: $d_{\text{loss}} =$ Wasserstein, $\Omega =$ Simplex.

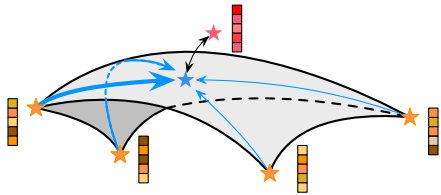
Gromov-Wasserstein Factorization for Graphs



- When $\Omega =$ Simplex, we have

$$\boldsymbol{A}\boldsymbol{\lambda} = \underbrace{\arg\min_{\boldsymbol{y}} \sum_{k=1}^{K} \lambda_k \|\boldsymbol{y} - \boldsymbol{a}_k\|_2^2}_{\text{Euclidean barycenter}} = \boldsymbol{b}(\boldsymbol{A}, \boldsymbol{\lambda}; \ell_2). \tag{76}$$

Gromov-Wasserstein Factorization for Graphs



- When $\Omega = \text{Simplex}$, we have

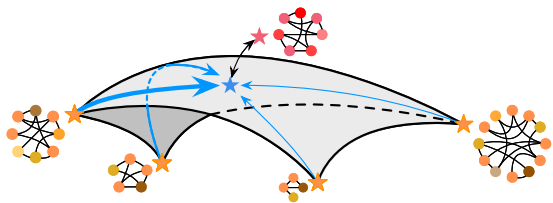
$$\boldsymbol{A}\boldsymbol{\lambda} = \underbrace{\arg\min_{\boldsymbol{y}} \sum_{k=1}^{K} \lambda_k \|\boldsymbol{y} - \boldsymbol{a}_k\|_2^2}_{\text{Euclidean barycenter}} = \boldsymbol{b}(\boldsymbol{A}, \boldsymbol{\lambda}; \ell_2). \tag{76}$$

- Accordingly, a generalized factorization model can be written as

$$\min_{\{\boldsymbol{A}, \boldsymbol{\lambda}_{1:I}\} \in \Omega} \sum_{i=1}^{I} d_{\text{loss}}(\boldsymbol{b}(\boldsymbol{A}, \boldsymbol{\lambda}_i; \underbrace{d_{\text{b}}}_{\text{b's metric}}), \boldsymbol{y}_i). \tag{77}$$

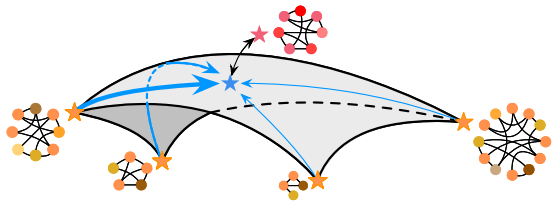
Gromov-Wasserstein Factorization for Graphs

Gromov-Wasserstein Factorization (GWF) [Xu, AAAI 2020]: Learning interpretable factorization model to represent unaligned graphs.



Gromov-Wasserstein Factorization for Graphs

Gromov-Wasserstein Factorization (GWF) [Xu, AAAI 2020]: Learning interpretable factorization model to represent unaligned graphs.



- Estimate each graph by a GW barycenter graph:

$$B_{gw}(\boldsymbol{U}_{1:K}, \boldsymbol{\lambda}) := \arg\min_{B} \sum\nolimits_{k=1}^{K} \lambda_k GW_2^2(B, G_k(\boldsymbol{U}_k)). \tag{78}$$

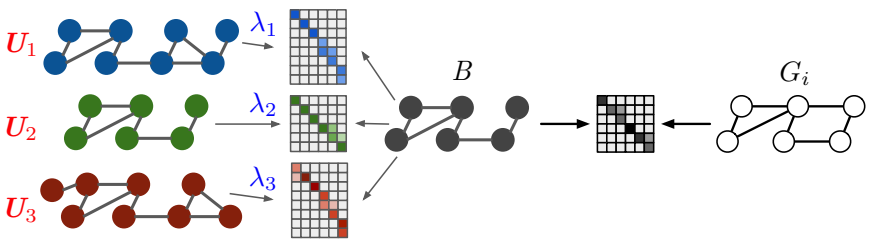
- $\{G_k(\boldsymbol{U}_k)\}_{k=1}^K$: a set of graph factors.
- $\boldsymbol{\lambda}_{1:I} = \{\boldsymbol{\lambda}_i \in \Delta^{K-1}\}_{i=1}^I$: the coefficients of the graph factors corresponding to $\{G_i\}_{i=1}^I$ (The representations of the observed graphs).

[Xu, AAAI 2020] Xu, H. Gromov-Wasserstein factorization models for graph clustering. AAAI 2020.

Learning Gromov-Wasserstein Factorization

Learning task:

$$\min_{\mathbf{1} \geq \boldsymbol{U}_{1:K} \geq \mathbf{0},\ \boldsymbol{\lambda}_{1:I} \in \Delta^{K-1}} \sum_{i=1}^{I} d_{\text{loss}}(B_{gw}(\boldsymbol{U}_{1:K}, \boldsymbol{\lambda}_i), G_i). \tag{79}$$



Learning Gromov-Wasserstein Factorization

Reparameterize the problem to an unconstrained optimization problem:

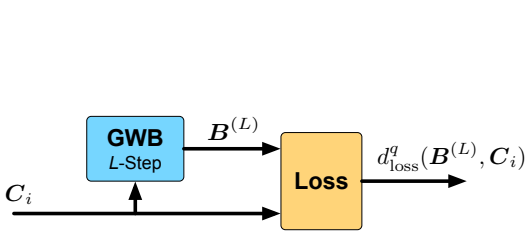
$$\min_{\boldsymbol{V}_{1:K},\ \boldsymbol{z}_{1:I}} \sum_{i=1}^{I} d_{\mathsf{loss}}(B_{gw}(\sigma(\boldsymbol{V}_{1:K}), \mathsf{softmax}(\boldsymbol{z}_i)), G_i). \tag{80}$$

Learning Gromov-Wasserstein Factorization

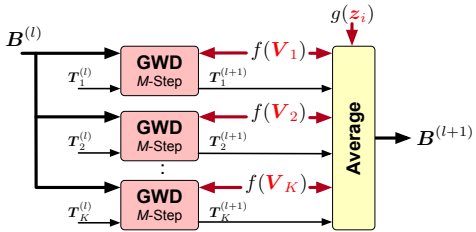
Reparameterize the problem to an unconstrained optimization problem:

$$\min_{\boldsymbol{V}_{1:K},\ \boldsymbol{z}_{1:I}} \sum_{i=1}^{I} d_{\text{loss}}(B_{gw}(\sigma(\boldsymbol{V}_{1:K}), \text{softmax}(\boldsymbol{z}_i)), G_i). \tag{80}$$

(1) Compute optimal transport matrices by GWD modules; (2) Fix the OT matrices and learn the model parameters via SGD:



Overall scheme

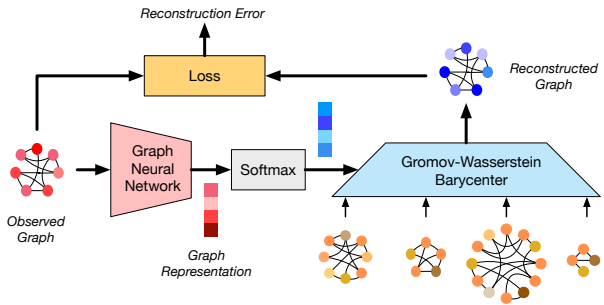


The GWB module

Parametrizing Coefficients Leads to A Graph Autoencoder

Parametrizing coefficients leads to an autoencoder (**GNN-GWF**) [Xu, et al, 2023]:

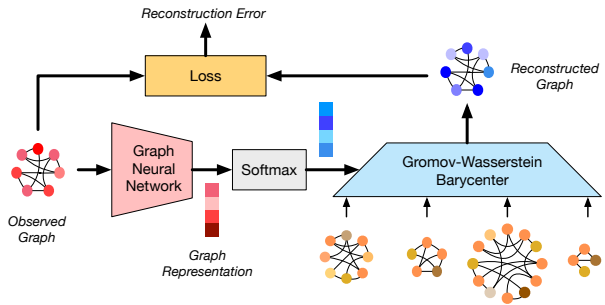
$$\min_{\boldsymbol{V}_{1:K},\ \boldsymbol{\theta}} \sum_{i=1}^{I} GW_2(B_{gw}(\sigma(\boldsymbol{V}_{1:K}), \text{softmax}(\text{GNN}_\theta(G_i))), G_i). \tag{81}$$



Parametrizing Coefficients Leads to A Graph Autoencoder

Parametrizing coefficients leads to an autoencoder (**GNN-GWF**) [Xu, et al, 2023]:

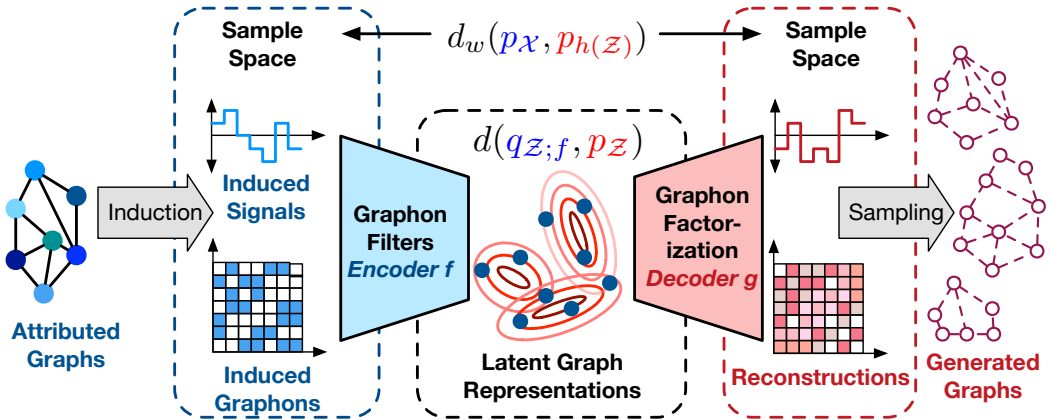
$$\min_{\boldsymbol{V}_{1:K},\ \boldsymbol{\theta}} \sum_{i=1}^{I} GW_2(B_{gw}(\sigma(\boldsymbol{V}_{1:K}), \text{softmax}(\text{GNN}_\theta(G_i))), G_i). \tag{81}$$



- Eq. (81) works as the reconstruction loss.
- Regularizing the distribution of $\text{GNN}_\theta(G)$ leads to a graph-level WAE.

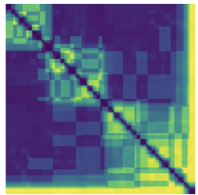
[Xu, et al, 2023] Representing graphs via Gromov-Wasserstein factorization. IEEE TPAMI, 2023

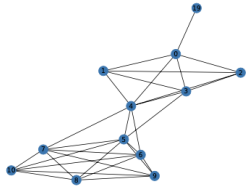
Extend to Graphon Wasserstein Autoencoder



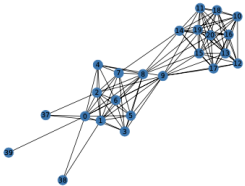
[Xu, et al, 2021] Xu, H., Zhao, P., Huang, J. and Luo, D., Learning Graphon Autoencoders for Generative Graph Modeling. arXiv:2105.14244. 2021.

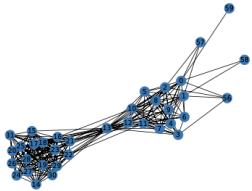
Preliminary Results for Transferable Graph Generation

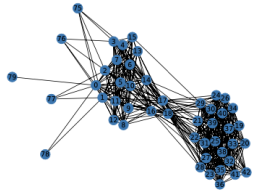




Decoded graphon and original graph







Generated graphs with different sizes

Summary

- Minimizing W_p is an important generative modeling paradigm, which can be implemented in different ways
  - Approximate it in its dual form
  - Approximate it in its primal form
  - Approximate it by (strongly or weakly) equivalent surrogates

Summary

- Minimizing W_p is an important generative modeling paradigm, which can be implemented in different ways
  - Approximate it in its dual form
  - Approximate it in its primal form
  - Approximate it by (strongly or weakly) equivalent surrogates
- Gromovize W_p leads to GW_p and its variants for structured data like graphs and point clouds.
  - The algorithms of W_p are applicable under slight modification
  - The problem becomes non-convex but the algorithms still lead to stationary points
  - GW barycenter provides a promising way to aggregate graphs

Summary

- Minimizing W_p is an important generative modeling paradigm, which can be implemented in different ways
  - Approximate it in its dual form
  - Approximate it in its primal form
  - Approximate it by (strongly or weakly) equivalent surrogates
- Gromovize W_p leads to GW_p and its variants for structured data like graphs and point clouds.
  - The algorithms of W_p are applicable under slight modification
  - The problem becomes non-convex but the algorithms still lead to stationary points
  - GW barycenter provides a promising way to aggregate graphs
- Applying GW distance and barycenter, we can develop generative models for graphs in different ways.
  - GWB-based graphon estimation
  - Gromov-Wasserstein factorization for graph representation and generation
  - Graph-level Wasserstein autoencoder

5-min break for Q & A

Outline

Data Privacy Issues in Machine Learning

- Case 1: I have data but they are noisy, incomplete, unaligned, ...
  - Keep robustness to imperfect data

Data Privacy Issues in Machine Learning

- Case 1: I have data but they are noisy, incomplete, unaligned, ...
  - Keep robustness to imperfect data
- Case 2: I don't have data because of privacy protection, limited budgets, poor sensors, ...
  - Achieve machine learning without raw data sharing

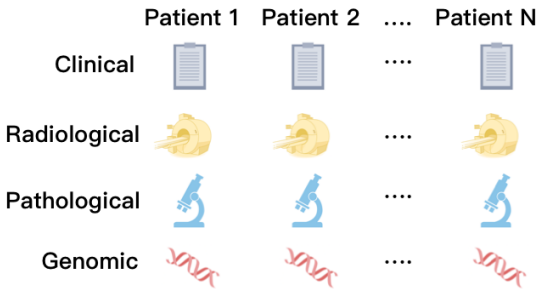
Data Privacy Issues in Machine Learning

- Case 1: I have data but they are noisy, incomplete, unaligned, ...
  - Keep robustness to imperfect data
- Case 2: I don't have data because of privacy protection, limited budgets, poor sensors, ...
  - Achieve machine learning without raw data sharing

We can develop OT-based solutions to the challenges.

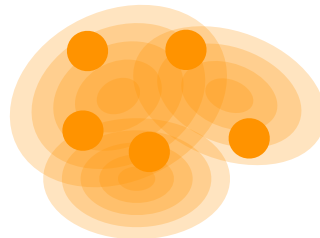
- OT-based robust multi-modal learning for Case 1
- Privacy-preserving distributed distribution comparison for Case 2

Two (Questionable) Assumptions on Multi-modal Learning



Well-aligned multi-modal data

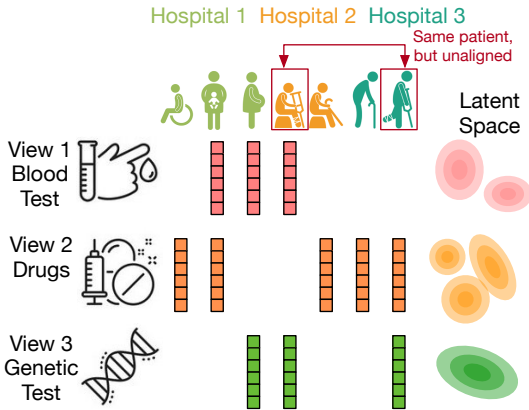
$$\mathbf{Z}_i \sim p_z \ \forall \ i$$



Shared latent distribution

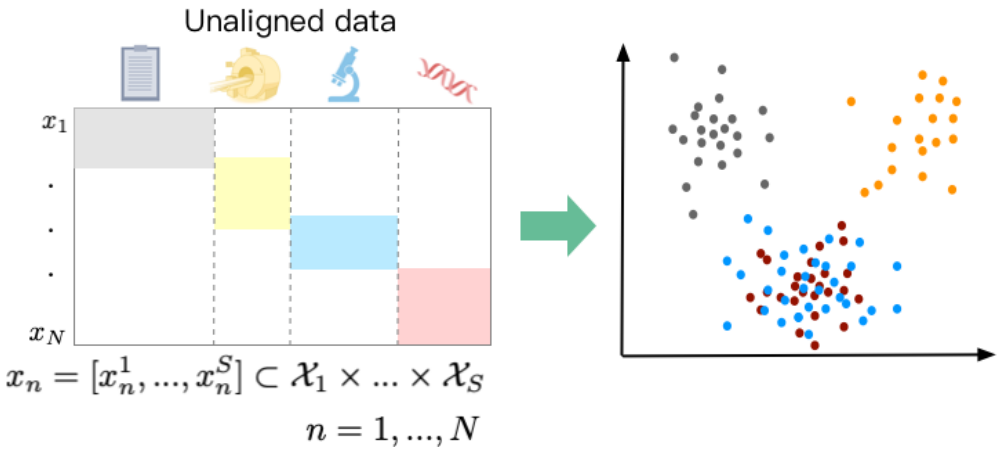
Real-world Multi-modal Scenarios

- Only do some tests
- Have admissions in different hospitals
- Collect data independently from different hospitals
- Complementary modalities



Real-world Multi-modal Scenarios

Unaligned multi-modal data + Clustered modalities in latent spaces.



Traditional Multi-modal Learning Paradigms

- Multi-modal data $[\boldsymbol{X}_1, ..., \boldsymbol{X}_S] \in \mathbb{R}^{N \times (D_1 + ... + D_S)}$.
- Learn latent representations directly or learn S encoders $\{f_s : \mathbb{R}^{D_s} \mapsto \mathcal{Z}\}_{s=1}^{S}$.

Traditional Multi-modal Learning Paradigms

- Multi-modal data $[\boldsymbol{X}_1, ..., \boldsymbol{X}_S] \in \mathbb{R}^{N \times (D_1 + ... + D_S)}$.
- Learn latent representations directly or learn S encoders $\{f_s : \mathbb{R}^{D_s} \mapsto \mathcal{Z}\}_{s=1}^{S}$.

Multi-kernel Fusion (MKF): Learn the encoders implicitly

$$\max_{\boldsymbol{U}, \{\alpha_s\}_{s=1}^{S}} \text{tr}(\boldsymbol{U}^\top \bar{\boldsymbol{K}} \boldsymbol{U}), \quad s.t.\ \bar{\boldsymbol{K}} = \sum_{s=1}^{S} \alpha_s \boldsymbol{K}_s. \tag{82}$$

Traditional Multi-modal Learning Paradigms

- Multi-modal data $[\boldsymbol{X}_1, ..., \boldsymbol{X}_S] \in \mathbb{R}^{N \times (D_1 + ... + D_S)}$.
- Learn latent representations directly or learn S encoders $\{f_s : \mathbb{R}^{D_s} \mapsto \mathcal{Z}\}_{s=1}^S$.

Multi-kernel Fusion (MKF): Learn the encoders implicitly

$$\max_{\boldsymbol{U}, \{\alpha_s\}_{s=1}^S} \text{tr}(\boldsymbol{U}^\top \bar{\boldsymbol{K}} \boldsymbol{U}), \quad s.t.\ \bar{\boldsymbol{K}} = \sum_{s=1}^S \alpha_s \boldsymbol{K}_s. \tag{82}$$

Canonical Correlation Analysis (CCA):

$$\begin{aligned} &\min_{\{f_s, U_s\}_{s=1}^S} \sum_{s \neq s'} \|U_s \circ f_s(\boldsymbol{X}_s) - U_{s'} \circ f_{s'}(\boldsymbol{X}_{s'})\|_F^2, \\ &s.t.\ (U_s \circ f_s(\boldsymbol{X}_s))^\top U_s \circ f_s(\boldsymbol{X}_s) = \boldsymbol{I},\ \forall s \end{aligned} \tag{83}$$

Traditional Multi-modal Learning Paradigms

- Multi-modal data $[\boldsymbol{X}_1, ..., \boldsymbol{X}_S] \in \mathbb{R}^{N \times (D_1 + ... + D_S)}$.
- Learn latent representations directly or learn S encoders $\{f_s : \mathbb{R}^{D_s} \mapsto \mathcal{Z}\}_{s=1}^S$.

Multi-kernel Fusion (MKF): Learn the encoders implicitly

$$\max_{\boldsymbol{U}, \{\alpha_s\}_{s=1}^S} \text{tr}(\boldsymbol{U}^\top \bar{\boldsymbol{K}} \boldsymbol{U}), \quad s.t.\ \bar{\boldsymbol{K}} = \sum_{s=1}^S \alpha_s \boldsymbol{K}_s. \tag{82}$$

Canonical Correlation Analysis (CCA):

$$\begin{aligned} &\min_{\{f_s, U_s\}_{s=1}^S} \sum_{s \neq s'} \|U_s \circ f_s(\boldsymbol{X}_s) - U_{s'} \circ f_{s'}(\boldsymbol{X}_{s'})\|_F^2, \\ &s.t.\ (U_s \circ f_s(\boldsymbol{X}_s))^\top U_s \circ f_s(\boldsymbol{X}_s) = \boldsymbol{I},\ \forall s \end{aligned} \tag{83}$$

Generalized Canonical Correlation Analysis (GCCA):

$$\min_{\{f_s, U_s\}_{s=1}^S, \boldsymbol{G}} \sum_{s=1}^S \|U_s \circ f_s(\boldsymbol{X}_s) - \boldsymbol{G}\|_F^2, \quad s.t.\ \boldsymbol{G}^\top \boldsymbol{G} = \boldsymbol{I},\ \forall s \tag{84}$$

Traditional Multi-modal Learning Paradigms

- Multi-modal data $[\boldsymbol{X}_1, ..., \boldsymbol{X}_S] \in \mathbb{R}^{N \times (D_1 + ... + D_S)}$.
- Learn latent representations directly or learn S encoders $\{f_s : \mathbb{R}^{D_s} \mapsto \mathcal{Z}\}_{s=1}^S$.

Multi-kernel Fusion (MKF): Learn the encoders implicitly

$$\max_{\boldsymbol{U}, \{\alpha_s\}_{s=1}^S} \text{tr}(\boldsymbol{U}^\top \bar{\boldsymbol{K}} \boldsymbol{U}), \quad s.t.\ \bar{\boldsymbol{K}} = \sum_{s=1}^S \alpha_s \boldsymbol{K}_s. \tag{82}$$

Canonical Correlation Analysis (CCA):

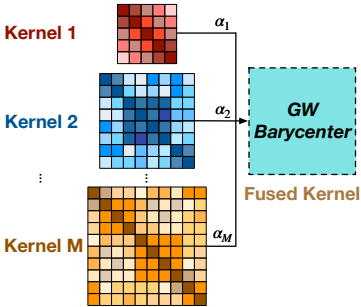
$$\begin{aligned} &\min_{\{f_s, U_s\}_{s=1}^S} \sum_{s \neq s'} \|U_s \circ f_s(\boldsymbol{X}_s) - U_{s'} \circ f_{s'}(\boldsymbol{X}_{s'})\|_F^2, \\ &s.t.\ (U_s \circ f_s(\boldsymbol{X}_s))^\top U_s \circ f_s(\boldsymbol{X}_s) = \boldsymbol{I},\ \forall s \end{aligned} \tag{83}$$

Generalized Canonical Correlation Analysis (GCCA):

$$\min_{\{f_s, U_s\}_{s=1}^S, \boldsymbol{G}} \sum_{s=1}^S \|U_s \circ f_s(\boldsymbol{X}_s) - \boldsymbol{G}\|_F^2, \quad s.t.\ \boldsymbol{G}^\top \boldsymbol{G} = \boldsymbol{I},\ \forall s \tag{84}$$

- How to make them applicable for unaligned data?
- How to introduce modality-level clustering structure?

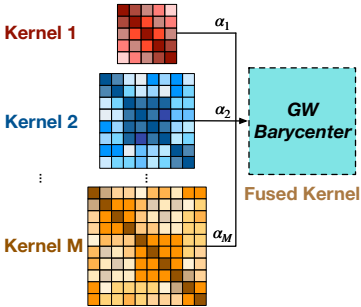
Extend MKF to Unaligned Data via GW Barycenters



Fuse kernels by solving a weighted GW barycenter problem:

$$\max_{\boldsymbol{U},\{\alpha_s\}_{s=1}^S} \mathrm{tr}(\boldsymbol{U}^\top \boldsymbol{K}\boldsymbol{U}),$$
$$s.t.\ \bar{\boldsymbol{K}} \in \min_{\boldsymbol{K}} \sum_{s=1}^S \alpha_s GW_2(\boldsymbol{K}, \boldsymbol{K}_s). \tag{85}$$

Extend MKF to Unaligned Data via GW Barycenters



Fuse kernels by solving a weighted GW barycenter problem:

$$\begin{aligned} &\max_{\boldsymbol{U},\{\alpha_s\}_{s=1}^S} \mathrm{tr}(\boldsymbol{U}^\top \boldsymbol{K}\boldsymbol{U}), \\ &s.t.\ \bar{\boldsymbol{K}} \in \min_{\boldsymbol{K}} \sum\nolimits_{s=1}^S \alpha_s GW_2(\boldsymbol{K}, \boldsymbol{K}_s). \end{aligned} \tag{85}$$

Nested optimization:

1. Compute the barycenter in a closed form

$$\bar{\boldsymbol{K}} = \frac{1}{S^2} \sum\nolimits_{s=1}^S \alpha_s \boldsymbol{T}_s^* \boldsymbol{K}_s (\boldsymbol{T}_s^*)^\top \tag{86}$$

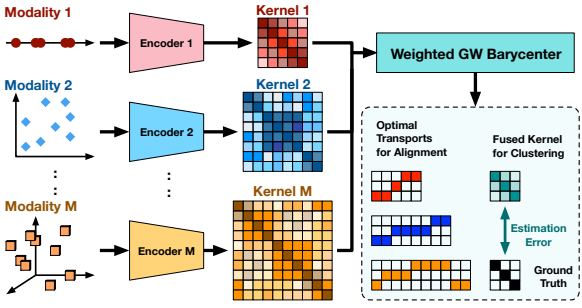
2. Plug the barycenter into the objective function:

$$\max_{\boldsymbol{U},\{\alpha_s\}_{s=1}^S} \mathrm{tr}\Big(\boldsymbol{U}^\top \Big(\sum\nolimits_{s=1}^S \alpha_s \boldsymbol{T}_s^* \boldsymbol{K}_s (\boldsymbol{T}_s^*)^\top \Big) \boldsymbol{U}\Big) \tag{87}$$

Extend MKF to Unaligned Data via GW Barycenters

When computing the kernels by latent codes, we obtain parametric kernels and the **Gromov-Wasserstein multi-modal alignment and clustering model:**

$$\underbrace{\max_{\boldsymbol{U}\in\Pi,\{f_s\}_{s=1}^S} \mathrm{tr}(\boldsymbol{U}^\top \boldsymbol{K}\boldsymbol{U})}_{\min GW(\bar{\boldsymbol{K}},\boldsymbol{I}_C)}, \quad s.t.\ \bar{\boldsymbol{K}} \in \min_{\boldsymbol{K}} \sum_{s=1}^S \alpha_s GW_2(\boldsymbol{K}, \underbrace{\boldsymbol{K}(f_s)}_{\text{param. kernel}}). \tag{88}$$



[Gong et al, 2022] Gong, F., Nie, Y. and Xu, H., Gromov-Wasserstein multi-modal alignment and clustering. CIKM, 2022.

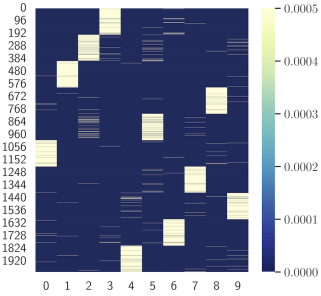
Multi-modal Clustering Performance

Table 2: The performance of different clustering methods. Here, "-" means that a method fails to obtain results in 10 hours.

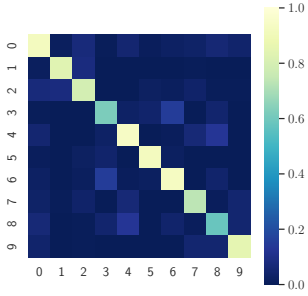
Data type	Datasets / Algorithms	HandWritten ACC	HandWritten NMI	Caltech 7 ACC	Caltech 7 NMI	ORL ACC	ORL NMI	Movies ACC	Movies NMI	Prokaryotic ACC	Prokaryotic NMI
Well-aligned ($\beta = 0$)	MCCA	0.8269	0.7775	**0.5313**	0.4716	0.3475	0.4992	0.0989	0.0722	0.5620	0.1204
	DCCAE	0.6537	0.6216	0.4110	0.3850	0.5625	0.7373	0.1572	0.1194	0.5070	0.1827
	AttnAE	0.7505	0.6912	0.4600	0.4575	0.4600	0.6603	0.1880	0.1918	0.5390	0.2625
	MVKSC	0.6749	0.6376	0.5196	0.2537	0.3013	0.5291	**0.2285**	**0.2098**	**0.6188**	**0.3191**
	MultiNMF	**0.8882**	**0.8279**	0.4525	**0.5120**	**0.6900**	**0.8100**	0.1726	0.1856	0.5771	0.2495
50% unaligned ($\beta = 0.5$)	CPM-GAN	0.7250	0.6069	0.3472	0.3151	0.1987	0.3703	0.1210	0.1753	0.3793	**0.3294**
	MVC-UM	-	-	**0.3958**	0.3838	**0.5863**	**0.7586**	0.1831	0.1950	0.3950	0.0807
	GWMAC	**0.8469**	**0.8156**	0.3541	**0.5010**	0.5322	0.7068	**0.1993**	**0.2195**	**0.5515**	0.3286
100% unaligned ($\beta = 1$)	MVC-UM	-	-	0.3112	0.2456	**0.5431**	**0.7452**	0.1841	0.1953	0.4451	0.0554
	GWMAC	**0.8144**	**0.7546**	**0.3568**	**0.4945**	0.5118	0.7026	**0.1928**	**0.2138**	**0.5479**	**0.3259**

Multi-modal Clustering Performance

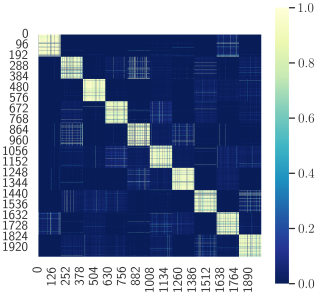
Visualize the clustering results of the Handwritten dataset



$T_s^* T_{obj}^*$



$\bar{K}$



$\sum_s K_s$

Extend CCA to Unaligned Data via Sliced Wasserstein

Sliced Wasserstein Canonical Correlation Analysis (SW-CCA):

$$\begin{aligned} &\min_{\{f_s, U_s\}_{s=1}^S} \sum\nolimits_{s \neq s'} \widehat{SW}_2^2(U_s \circ f_s(\boldsymbol{X}_s), U_{s'} \circ f_{s'}(\boldsymbol{X}_{s'})), \\ &s.t.\ (U_s \circ f_s(\boldsymbol{X}_s))^\top U_s \circ f_s(\boldsymbol{X}_s) = \boldsymbol{I},\ \forall s \end{aligned} \tag{89}$$

Sliced Wasserstein Generalized Canonical Correlation Analysis (SW-GCCA):

$$\min_{\{f_s, U_s\}_{s=1}^S, \boldsymbol{G}} \sum_{s=1}^S \widehat{SW}_2^2(U_s \circ f_s(\boldsymbol{X}_s), \boldsymbol{G}), \quad s.t.\ \boldsymbol{G}^\top \boldsymbol{G} = \boldsymbol{I},\ \forall s \tag{90}$$

Extend CCA to Unaligned Data via Sliced Wasserstein

Sliced Wasserstein Canonical Correlation Analysis (SW-CCA):

$$\begin{aligned} &\min_{\{f_s,U_s\}_{s=1}^S} \sum\nolimits_{s\neq s'} \widehat{SW}_2^2(U_s\circ f_s(\boldsymbol{X}_s), U_{s'}\circ f_{s'}(\boldsymbol{X}_{s'})), \\ &s.t.\ (U_s\circ f_s(\boldsymbol{X}_s))^\top U_s\circ f_s(\boldsymbol{X}_s)=\boldsymbol{I},\ \forall s \end{aligned} \tag{89}$$

Sliced Wasserstein Generalized Canonical Correlation Analysis (SW-GCCA):

$$\min_{\{f_s,U_s\}_{s=1}^S,\boldsymbol{G}} \sum_{s=1}^S \widehat{SW}_2^2(U_s\circ f_s(\boldsymbol{X}_s),\boldsymbol{G}),\quad s.t.\ \boldsymbol{G}^\top\boldsymbol{G}=\boldsymbol{I},\ \forall s \tag{90}$$

- Using SW distance does not require aligned data.
- It is differentiable and efficient, just requiring random projections and sorting operations.

[Luo et al., 2022] Luo, D., Xu, H. and Carin, L., Differentiable Hierarchical Optimal Transport for Robust Multi-View Learning. TPAMI, 2022.

Extend CCA to Unaligned Data via Sliced Wasserstein

- Treat U_s as a linear random projector, i.e., $U_s : \mathcal{Z} \mapsto \mathbb{R}$, and learn it in an adversarial way, we have

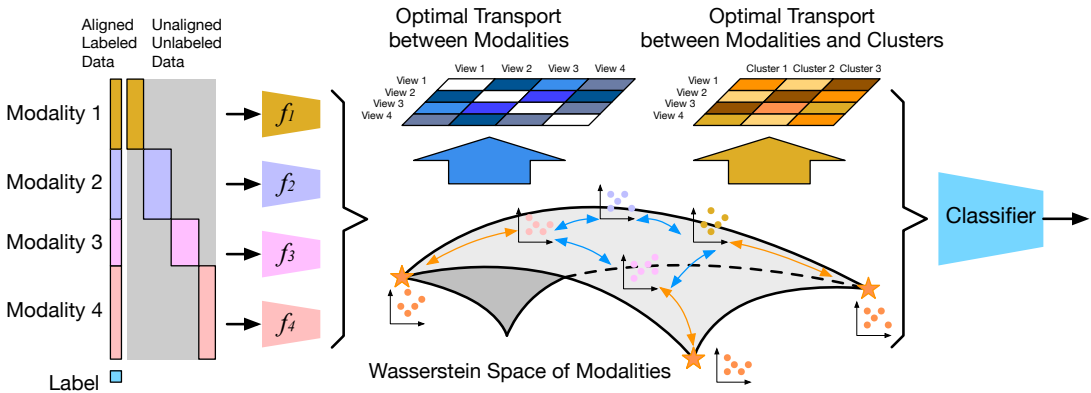
Max-Sliced Wasserstein Canonical Correlation Analysis (MSW-CCA):

$$\begin{aligned} &\min_{\{f_s\}_{s=1}^S} \sum\nolimits_{s \neq s'} \widehat{MSW}_2^2(f_s(\boldsymbol{X}_s), f_{s'}(\boldsymbol{X}_{s'})), \\ &s.t.\ (U_s \circ f_s(\boldsymbol{X}_s))^\top U_s \circ f_s(\boldsymbol{X}_s) = \boldsymbol{I},\ \forall s \end{aligned} \tag{91}$$

Max-Sliced Wasserstein Generalized Canonical Correlation Analysis (MSW-GCCA):

$$\min_{\{f_s\}_{s=1}^S, \boldsymbol{G}} \sum_{s=1}^S \widehat{MSW}_2^2(f_s(\boldsymbol{X}_s), \boldsymbol{G}), \quad s.t.\ \boldsymbol{G}^\top \boldsymbol{G} = \boldsymbol{I},\ \forall s \tag{92}$$

Hierarchical Optimal Transport for Modality Clustering



Principle:

- Further extend SW-CCA and SW-GCCA
- Capture the relations among the modalities by their OT distances.

Hierarchical Optimal Transport for Modality Clustering

Extend SW-CCA: Learn the pairwise relations between different modalities.

$$\min_{\substack{\{f_s,U_s\}_{s=1}^S,\\ \boldsymbol{W}\in\Pi(\frac{1}{S}1_S,\frac{1}{S}1_S)}} \underbrace{\sum\nolimits_{s\neq s'} w_{ss'}\widehat{SW}_2^2(U_s\circ f_s(\boldsymbol{X}_s),U_{s'}\circ f_{s'}(\boldsymbol{X}_{s'}))}_{\text{Hierarchical OT}} + \alpha\underbrace{\Big\|\sum\nolimits_s (U_s\circ f_s(\boldsymbol{X}_s))^\top U_s\circ f_s(\boldsymbol{X}_s)-\boldsymbol{I}\Big\|_F^2}_{\text{CCA-Regularizer}} + \beta\underbrace{H(\boldsymbol{W})}_{\langle \boldsymbol{W},\log \boldsymbol{W}\rangle}. \tag{93}$$

- **Lower level:** the SW distance between different modalities' sample sets.
- **Upper level:** Take the SW distances as the cost matrix, compute the EOT between the group of modalities and itself. (Set $w_{ss}=0$ to avoid trivial solutions)

Hierarchical Optimal Transport for Modality Clustering

Extend SW-CCA: Learn the pairwise relations between different modalities.

$$\min_{\substack{\{f_s,U_s\}_{s=1}^S,\\ \boldsymbol{W}\in\Pi(\frac{1}{S}1_S,\frac{1}{S}1_S)}} \underbrace{\sum\nolimits_{s\neq s'} w_{ss'}\widehat{SW}_2^2(U_s\circ f_s(\boldsymbol{X}_s),U_{s'}\circ f_{s'}(\boldsymbol{X}_{s'}))}_{\text{Hierarchical OT}} + \alpha\underbrace{\Big\|\sum\nolimits_s(U_s\circ f_s(\boldsymbol{X}_s))^\top U_s\circ f_s(\boldsymbol{X}_s)-\boldsymbol{I}\Big\|_F^2}_{\text{CCA-Regularizer}}+\beta\underbrace{H(\boldsymbol{W})}_{\langle \boldsymbol{W},\log \boldsymbol{W}\rangle}. \tag{93}$$

- **Lower level:** the SW distance between different modalities' sample sets.
- **Upper level:** Take the SW distances as the cost matrix, compute the EOT between the group of modalities and itself. (Set $w_{ss}=0$ to avoid trivial solutions)
- $\boldsymbol{W}^*$ indicates the clustering structure implicitly by the pairwise similarity between different modalities.

Hierarchical Optimal Transport for Modality Clustering

Extend SW-GCCA: Introduce K learnable reference matrix $\{\boldsymbol{G}_k\}_{k=1}^K$ and learn the pairwise relations between the modalities and the references.

$$\min_{\substack{\{f_s,U_s\}_{s=1}^S,\{\boldsymbol{G}_k\}_{k=1}^K,\\ \boldsymbol{W}\in\Pi(\frac{1}{S}1_S,\frac{1}{K}1_K)}} \underbrace{\sum\nolimits_{s,k} w_{sk}\widehat{SW}_2^2(U_s\circ f_s(X_s),\boldsymbol{G}_k)}_{\text{Hierarchical OT}} + \alpha\underbrace{\left\|\sum\nolimits_k \boldsymbol{G}_k\boldsymbol{G}_k^\top-\boldsymbol{I}\right\|_F^2}_{\text{GCCA Regularizer}}+\beta H(\boldsymbol{W}). \tag{94}$$

- **Lower level:** the SW distance between each modality's sample set and the reference.
- **Upper level:** Take the SW distances as the cost matrix, compute the EOT between the group of modalities and the clusters.

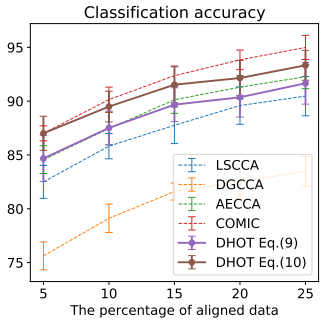
Hierarchical Optimal Transport for Modality Clustering

Extend SW-GCCA: Introduce K learnable reference matrix $\{\boldsymbol{G}_k\}_{k=1}^K$ and learn the pairwise relations between the modalities and the references.

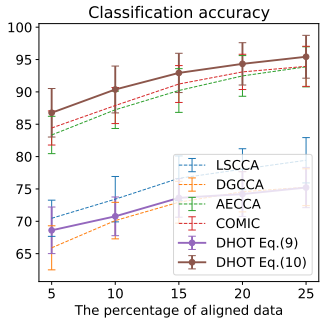
$$\min_{\substack{\{f_s,U_s\}_{s=1}^S,\{\boldsymbol{G}_k\}_{k=1}^K,\\ \boldsymbol{W}\in\Pi(\frac{1}{S}1_S,\frac{1}{K}1_K)}} \underbrace{\sum\nolimits_{s,k} w_{sk}\widehat{SW}_2^2(U_s\circ f_s(X_s),\boldsymbol{G}_k)}_{\text{Hierarchical OT}} + \alpha\underbrace{\left\|\sum\nolimits_k \boldsymbol{G}_k\boldsymbol{G}_k^\top-\boldsymbol{I}\right\|_F^2}_{\text{GCCA Regularizer}}+\beta H(\boldsymbol{W}). \tag{94}$$

- **Lower level:** the SW distance between each modality's sample set and the reference.
- **Upper level:** Take the SW distances as the cost matrix, compute the EOT between the group of modalities and the clusters.
- $\boldsymbol{W}^*$ indicates the clustering structure explicitly.

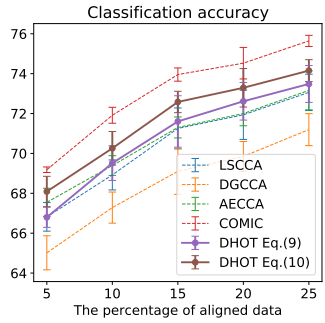
Performance on Multi-modal Classification



Caltech7



Handwritten



Cathgen

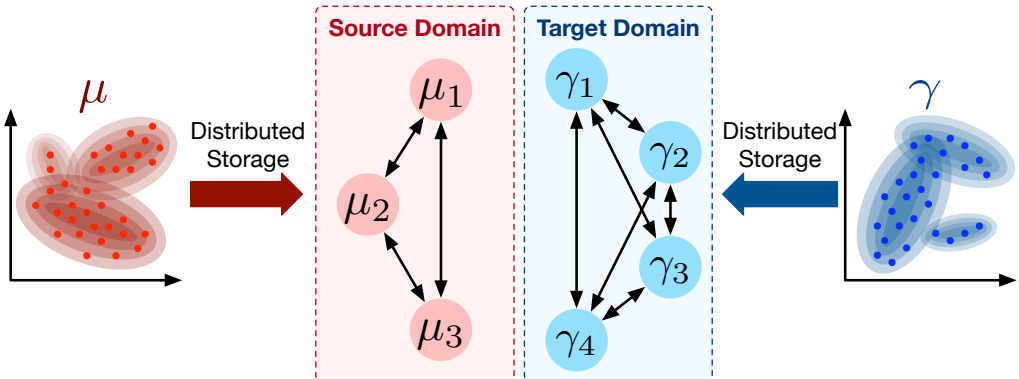
[Luo et al., 2022] Luo, D., Xu, H. and Carin, L., Differentiable Hierarchical Optimal Transport for Robust Multi-View Learning. TPAMI, 2022.

Distributed Distribution Comparison

- In all above work, the data of each distribution are assumed to be stored in a centralized way. The accessibility of the data is not considered.

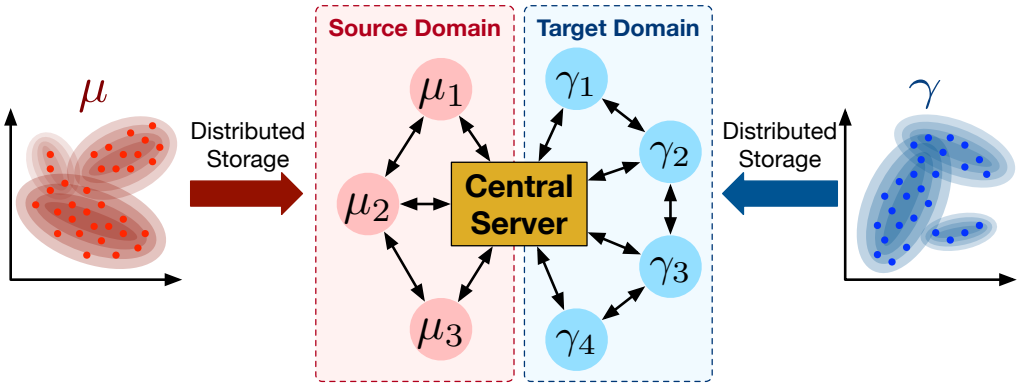
Distributed Distribution Comparison

- In all above work, the data of each distribution are assumed to be stored in a centralized way. The accessibility of the data is not considered.



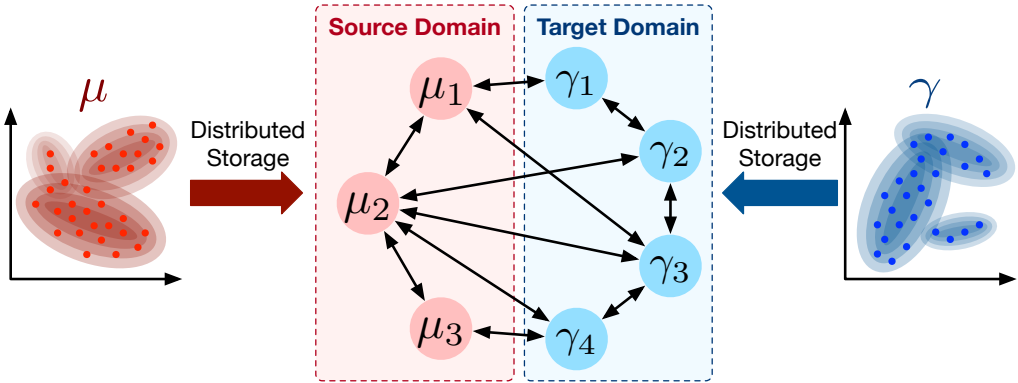
- Real-world large-scale data are often stored in different servers in a distributed way.
- We need to find a way to achieve **distributed distribution comparison**.

(Centralized) Distributed Distribution Comparison



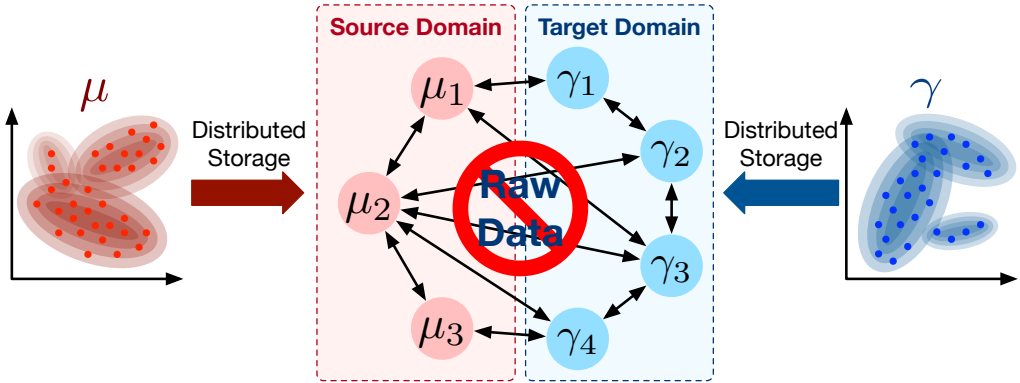
- High communication cost or high communication bandwidth requirement
- High storage cost in the central server.

(Decentralized) Distributed Distribution Comparison



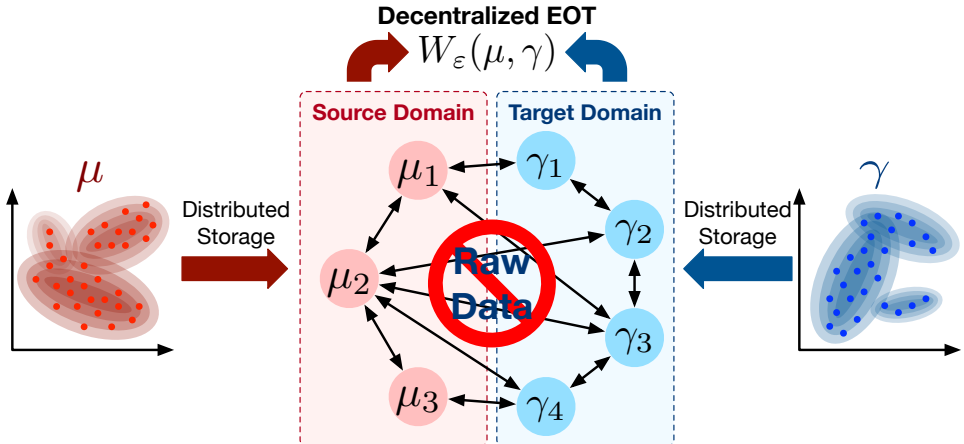
- Low cost on storage and low bandwidth cost
- Slow convergence (maybe high communication cost)

Privacy-preserving Distributed Distribution Comparison



- Approximation precision + privacy preservation + communication efficiency
- Distributed Domain Adaptation, Federated Learning, ...

A Potential Solution: Decentralized Entropic Optimal Transport



Entropic Optimal Transport and Its Fenchel Dual Form

Entropic optimal transport (EOT) distance, or called Sinkhorn distance:

$$W_\varepsilon(\mu,\gamma) := \Big(\inf_{\pi\in\Pi(\mu,\gamma)} \int_{\mathcal{X}^2} \underbrace{c(x,y)}_{\|x-y\|_p^p} \mathrm{d}\pi(x,y)\Big)^{1/p} - \varepsilon H(\pi), \tag{95}$$

Entropic Optimal Transport and Its Fenchel Dual Form

Entropic optimal transport (EOT) distance, or called Sinkhorn distance:

$$W_\varepsilon(\mu,\gamma) := \Big(\inf_{\pi\in\Pi(\mu,\gamma)} \int_{\mathcal{X}^2} \underbrace{c(x,y)}_{\|x-y\|_p^p} \mathrm{d}\pi(x,y)\Big)^{1/p} - \varepsilon H(\pi), \tag{95}$$

- **Strictly-convex**, but $c(x,y)$ and $\pi(x,y)$ are not friendly for decentralized cases.

Entropic Optimal Transport and Its Fenchel Dual Form

Entropic optimal transport (EOT) distance, or called Sinkhorn distance:

$$W_\varepsilon(\mu,\gamma) := \Big(\inf_{\pi\in\Pi(\mu,\gamma)} \int_{\mathcal{X}^2} \underbrace{c(x,y)}_{\|x-y\|_p^p} \mathrm{d}\pi(x,y)\Big)^{1/p} - \varepsilon H(\pi), \tag{95}$$

- **Strictly-convex**, but $c(x,y)$ and $\pi(x,y)$ are not friendly for decentralized cases.

Fenchel Dual Form of EOT:

$$\begin{aligned} W_\varepsilon^p(\mu,\gamma) &:= \sup_{u,v\in\mathcal{C}_\mathcal{X}} \int_\mathcal{X} u(x)\mathrm{d}\mu(x) + \int_\mathcal{X} v(y)\mathrm{d}\gamma(y) - \varepsilon \int_{\mathcal{X}^2} e^{\frac{u(x)+v(y)-c(x,y)}{\varepsilon}} \mathrm{d}\mu(x)\mathrm{d}\gamma(y) \\ &= \sup_{u,v\in\mathcal{C}_\mathcal{X}} \mathbb{E}_{x\sim\mu,y\sim\gamma}[\underbrace{u(x)+v(y)-\varepsilon e^{\frac{u(x)+v(y)}{\varepsilon}} \overbrace{e^{-\frac{c(x,y)}{\varepsilon}}}^{\kappa(x,y)}}_{f_\varepsilon(x,y,u,v)}] \end{aligned} \tag{96}$$

- **Unconstrained** dual functions + **Expectation**-based formulation

Entropic Optimal Transport and Its Fenchel Dual Form

Entropic optimal transport (EOT) distance, or called Sinkhorn distance:

$$W_\varepsilon(\mu,\gamma) := \Big(\inf_{\pi\in\Pi(\mu,\gamma)} \int_{\mathcal{X}^2} \underbrace{c(x,y)}_{\|x-y\|_p^p} \mathrm{d}\pi(x,y)\Big)^{1/p} - \varepsilon H(\pi), \tag{95}$$

- **Strictly-convex**, but $c(x,y)$ and $\pi(x,y)$ are not friendly for decentralized cases.

Fenchel Dual Form of EOT:

$$\begin{aligned} W_\varepsilon^p(\mu,\gamma) &:= \sup_{u,v\in\mathcal{C}_\mathcal{X}} \int_\mathcal{X} u(x)\mathrm{d}\mu(x) + \int_\mathcal{X} v(y)\mathrm{d}\gamma(y) - \varepsilon \int_{\mathcal{X}^2} e^{\frac{u(x)+v(y)-c(x,y)}{\varepsilon}} \mathrm{d}\mu(x)\mathrm{d}\gamma(y) \\ &= \sup_{u,v\in\mathcal{C}_\mathcal{X}} \mathbb{E}_{x\sim\mu,y\sim\gamma}[\underbrace{u(x)+v(y)-\varepsilon e^{\frac{u(x)+v(y)}{\varepsilon}} \overbrace{e^{-\frac{c(x,y)}{\varepsilon}}}^{\kappa(x,y)}}_{f_\varepsilon(x,y,u,v)}] \end{aligned} \tag{96}$$

- **Unconstrained** dual functions + **Expectation**-based formulation

Keypoint: A privacy-preserving and communication-efficient approximation of $\kappa(x,y)$.

The Dual Problem in Decentralized Scenarios

- **Storage protocol** $p \otimes q = [p_i q_j]$**:**
  - **Hierarchical model:** Select agents via p or q. Scatter data via μ_i or γ_j.

The Dual Problem in Decentralized Scenarios

- **Storage protocol** $p \otimes q = [p_i q_j]$**:**
  - **Hierarchical model:** Select agents via p or q. Scatter data via μ_i or γ_j.

$$\mu = \sum_{i=1}^{I} p_i \mu_i, \quad \gamma = \sum_{j=1}^{J} q_j \gamma_j. \tag{97}$$

The Dual Problem in Decentralized Scenarios

- **Storage protocol** $p \otimes q = [p_i q_j]$**:**
  - **Hierarchical model:** Select agents via p or q. Scatter data via μ_i or γ_j.

$$\mu = \sum_{i=1}^{I} p_i \mu_i, \quad \gamma = \sum_{j=1}^{J} q_j \gamma_j. \tag{97}$$

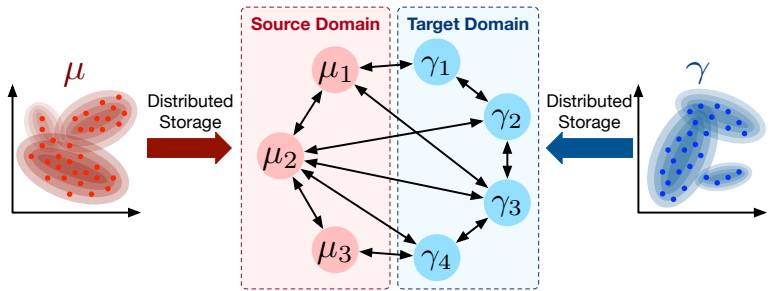
- **Communication protocol** $E = [e_{ij}]$**:**
  - e_{ij} is the probability of the source agent i communicates with the target agent j.

The Dual Problem in Decentralized Scenarios

- **Storage protocol** $p \otimes q = [p_i q_j]$**:**
  - **Hierarchical model:** Select agents via p or q. Scatter data via μ_i or γ_j.

$$\mu = \sum_{i=1}^{I} p_i \mu_i, \quad \gamma = \sum_{j=1}^{J} q_j \gamma_j. \tag{97}$$

- **Communication protocol** $E = [e_{ij}]$**:**
  - e_{ij} is the probability of the source agent i communicates with the target agent j.



The Dual Problem in Decentralized Scenarios

The **oracle** under storage protocol:

$$W_\varepsilon(\mu,\gamma) = \sup_{u,v\in\mathcal{C}_\mathcal{X}} \mathbb{E}_{x\sim\mu,y\sim\gamma} f_\varepsilon(x,y,u,v)$$

The Dual Problem in Decentralized Scenarios

The **oracle** under storage protocol:

$$\begin{aligned} W_\varepsilon(\mu,\gamma) &= \sup_{u,v\in\mathcal{C}_\mathcal{X}} \mathbb{E}_{x\sim\mu,y\sim\gamma} f_\varepsilon(x,y,u,v) \\ &= \sup_{u,v\in\mathcal{C}_\mathcal{X}} \mathbb{E}_{(i,j)\sim p\otimes q}\mathbb{E}_{x\sim\mu_i,y\sim\gamma_j} f_\varepsilon(x,y,u,v). \end{aligned} \tag{98}$$

The Dual Problem in Decentralized Scenarios

The **oracle** under storage protocol:

$$\begin{aligned} W_\varepsilon(\mu,\gamma) &= \sup_{u,v\in\mathcal{C}_\mathcal{X}} \mathbb{E}_{x\sim\mu,y\sim\gamma} f_\varepsilon(x,y,u,v) \\ &= \sup_{u,v\in\mathcal{C}_\mathcal{X}} \mathbb{E}_{(i,j)\sim p\otimes q} \mathbb{E}_{x\sim\mu_i,y\sim\gamma_j} f_\varepsilon(x,y,u,v). \end{aligned} \tag{98}$$

The **real case** under communication protocol:

$$\widetilde{W}_\varepsilon(\mu,\gamma) = \sup_{u,v\in\mathcal{C}_\mathcal{X}} \mathbb{E}_{(i,j)\sim E} \mathbb{E}_{x\sim\mu_i,y\sim\gamma_j} f_\varepsilon(x,y,u,v). \tag{99}$$

The Dual Problem in Decentralized Scenarios

The **oracle** under storage protocol:

$$\begin{aligned} W_\varepsilon(\mu,\gamma) &= \sup_{u,v\in\mathcal{C}_\mathcal{X}} \mathbb{E}_{x\sim\mu,y\sim\gamma} f_\varepsilon(x,y,u,v) \\ &= \sup_{u,v\in\mathcal{C}_\mathcal{X}} \mathbb{E}_{(i,j)\sim p\otimes q}\mathbb{E}_{x\sim\mu_i,y\sim\gamma_j} f_\varepsilon(x,y,u,v). \end{aligned} \tag{98}$$

The **real case** under communication protocol:

$$\widetilde{W}_\varepsilon(\mu,\gamma) = \sup_{u,v\in\mathcal{C}_\mathcal{X}} \mathbb{E}_{(i,j)\sim E}\mathbb{E}_{x\sim\mu_i,y\sim\gamma_j} f_\varepsilon(x,y,u,v). \tag{99}$$

Theorem (Protocol Mismatching Error)

When $\max_{i,j} W_\varepsilon(\mu_i,\gamma_j) \le \tau$ *and* $\sum_{i,j}|e_{ij}-p_iq_j| \le \sigma$,

$$|\widetilde{W}_\varepsilon(\mu,\gamma) - W_\varepsilon(\mu,\gamma)| \le \tau\sigma. \tag{100}$$

Sample-based Optimization

Given N source samples and M target samples, we have

$$\max_{\substack{u=\{u^{(i)}\}_{i=1}^{I}\in\mathbb{R}^{N}\\ v=\{v^{(j)}\}_{j=1}^{J}\in\mathbb{R}^{M}}} \overbrace{\sum_{i=1}^{I}\sum_{j=1}^{J}\frac{e_{ij}}{N_i M_j}\underbrace{f_\varepsilon(K_{ij},u^{(i)},v^{(j)})}_{f_\varepsilon^{(i,j)}}}^{F_\varepsilon(u,v;K,E)}. \tag{101}$$

$K=[\kappa(x_n,y_m)]\in\mathbb{R}^{N\times M}$ and $K_{ij}=[\kappa(x_n^{(i)},y_m^{(j)})]\in\mathbb{R}^{N_i\times M_j}$ is a block of K.

Sample-based Optimization

Given N source samples and M target samples, we have

$$\max_{\substack{u=\{u^{(i)}\}_{i=1}^{I}\in\mathbb{R}^{N}\\ v=\{v^{(j)}\}_{j=1}^{J}\in\mathbb{R}^{M}}} \overbrace{\sum_{i=1}^{I}\sum_{j=1}^{J}\frac{e_{ij}}{N_i M_j}\underbrace{f_\varepsilon(K_{ij},u^{(i)},v^{(j)})}_{f_\varepsilon^{(i,j)}}}^{F_\varepsilon(u,v;K,E)}. \tag{101}$$

$K=[\kappa(x_n,y_m)]\in\mathbb{R}^{N\times M}$ and $K_{ij}=[\kappa(x_n^{(i)},y_m^{(j)})]\in\mathbb{R}^{N_i\times M_j}$ is a block of K.

- The decomposable global dual objective F_ε:
  - Each local dual objective $f_\varepsilon^{(i,j)}$ involves an agent pair.
  - The dual variables can be scattered to different agents.
  - The global kernel can be one-step and blockwisely computed.

Sample-based Optimization

Given N source samples and M target samples, we have

$$\max_{\substack{u=\{u^{(i)}\}_{i=1}^{I}\in\mathbb{R}^{N}\\ v=\{v^{(j)}\}_{j=1}^{J}\in\mathbb{R}^{M}}} \overbrace{\sum_{i=1}^{I}\sum_{j=1}^{J}\frac{e_{ij}}{N_i M_j}\underbrace{f_\varepsilon(K_{ij},u^{(i)},v^{(j)})}_{f_\varepsilon^{(i,j)}}}^{F_\varepsilon(u,v;K,E)}. \tag{101}$$

$K=[\kappa(x_n,y_m)]\in\mathbb{R}^{N\times M}$ and $K_{ij}=[\kappa(x_n^{(i)},y_m^{(j)})]\in\mathbb{R}^{N_i\times M_j}$ is a block of K.

- The decomposable global dual objective F_ε:
  - Each local dual objective $f_\varepsilon^{(i,j)}$ involves an agent pair.
  - The dual variables can be scattered to different agents.
  - The global kernel can be one-step and blockwisely computed.
- The keypoint is approximating each local kernel K_{ij}.

Privacy-preserving and Communication-efficient Kernel Estimation

- Treat $\kappa(x, y)$ as a kind of generalized inner product (GIP):

$$\kappa(x, y) = \exp(-\|x - y\|_2^2/\varepsilon) = \underbrace{g(\theta_{\langle x,y\rangle}, \|x\|, \|y\|)}_{G\text{-Lipschitz}}. \tag{102}$$

Privacy-preserving and Communication-efficient Kernel Estimation

- Treat $\kappa(x, y)$ as a kind of generalized inner product (GIP):

$$\kappa(x, y) = \exp(-\|x - y\|_2^2/\varepsilon) = \underbrace{g(\theta_{\langle x,y\rangle}, \|x\|, \|y\|)}_{G\text{-Lipschitz}}. \tag{102}$$

- Given N_i samples $\{x_n^{(i)}\}_{n=1}^{N_i}$ and P random vectors $\{\omega_\ell\}_{\ell=1}^{P}$, we have

$$A_{\mu_i} = [\mathbb{I}(\langle \omega_\ell, x_n^{(i)} \rangle \geq 0)] \in \{0, 1\}^{P \times N_i},$$

Privacy-preserving and Communication-efficient Kernel Estimation

- Treat $\kappa(x, y)$ as a kind of generalized inner product (GIP):

$$\kappa(x, y) = \exp(-\|x - y\|_2^2/\varepsilon) = \underbrace{g(\theta_{\langle x,y\rangle}, \|x\|, \|y\|)}_{G\text{-Lipschitz}}. \tag{102}$$

- Given N_i samples $\{x_n^{(i)}\}_{n=1}^{N_i}$ and P random vectors $\{\omega_\ell\}_{\ell=1}^{P}$, we have

$$\begin{aligned} A_{\mu_i} &= [\mathbb{I}(\langle \omega_\ell, x_n^{(i)} \rangle \geq 0)] \in \{0, 1\}^{P \times N_i}, \\ \hat{\kappa}(x_n^{(i)}, y_m^{(j)}) &= g\Big(\pi \left|1 - \frac{2}{P}\langle a_n^{(i)}, a_m^{(j)} \rangle\right|, \|x_n^{(i)}\|, \|y_m^{(j)}\|\Big), \end{aligned} \tag{103}$$

Privacy-preserving and Communication-efficient Kernel Estimation

- Treat $\kappa(x, y)$ as a kind of generalized inner product (GIP):

$$\kappa(x, y) = \exp(-\|x - y\|_2^2/\varepsilon) = \underbrace{g(\theta_{\langle x,y\rangle}, \|x\|, \|y\|)}_{G\text{-Lipschitz}}. \tag{102}$$

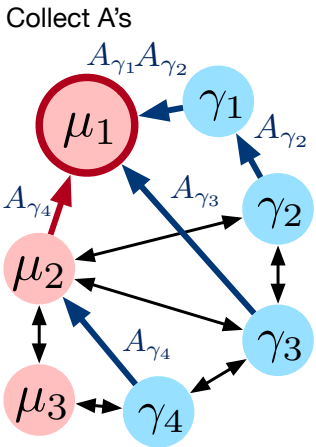
- Given N_i samples $\{x_n^{(i)}\}_{n=1}^{N_i}$ and P random vectors $\{\omega_\ell\}_{\ell=1}^{P}$, we have

$$\begin{aligned} A_{\mu_i} &= [\mathbb{I}(\langle \omega_\ell, x_n^{(i)} \rangle \geq 0)] \in \{0,1\}^{P \times N_i}, \\ \hat{\kappa}(x_n^{(i)}, y_m^{(j)}) &= g\Big(\pi \left|1 - \frac{2}{P}\langle a_n^{(i)}, a_m^{(j)} \rangle\right|, \|x_n^{(i)}\|, \|y_m^{(j)}\|\Big), \end{aligned} \tag{103}$$

Theorem (Kernel Approx. Error, based on (Khanduri, et al., ICLR 2021))

$$\mathbb{P}\left(\|K - \widehat{K}\| \leq G(N+M)\Big(\sqrt{\frac{32\pi^2}{P}\log\frac{2(N+M)}{\delta}} + \frac{8\pi}{3P}\log\frac{2(N+M)}{\delta}\Big)\right) \geq 1 - \delta \tag{104}$$

Step 1 of DEOT: Compute and Broadcast A's



1: Draw random variables $\{\omega_\ell \in \mathbb{R}^D\}_{\ell=1}^P \sim \mathcal{N}(0, I_D)$ and broadcast them to all agents.
2: **for** Each source agent $i \in \{1, ..., I\}$ **do**
3: Construct A_{μ_i} via (11) and broadcast it to all target agents. $\mathcal{O}(JN_iP)$
4: If data is not normalized, broadcast $\{\|x_n^{(i)}\|\}_{n=1}^{N_i}$ to all target agents. $\mathcal{O}(JN_i)$
5: **end for**
6: **for** Each target agent $j \in \{1, ..., J\}$ **do**
7: Construct A_{γ_j} via (11) and broadcast it to all source agents. $\mathcal{O}(IM_jP)$
8: If data is not normalized, broadcast $\{\|y_m^{(j)}\|\}_{m=1}^{M_j}$ to all source agents. $\mathcal{O}(IM_j)$
9: **end for**

Step 2 of DEOT: Update Dual Variables via MRBCD

- Privacy-preserving objective: $F_\varepsilon(u, v; \widehat{K}, E) = \sum_{i,j} e_{ij} \hat{f}_\varepsilon^{(i,j)}$, where $\hat{f}_\varepsilon^{(i,j)} = f_\varepsilon(\widehat{K}_{ij}, u^{(i)}, v^{(j)})$.
- Apply mini-batch randomized block coordinate descent (MRBCD).

Step 2 of DEOT: Update Dual Variables via MRBCD

- Privacy-preserving objective: $F_\varepsilon(u, v; \widehat{K}, E) = \sum_{i,j} e_{ij} \hat{f}_\varepsilon^{(i,j)}$, where $\hat{f}_\varepsilon^{(i,j)} = f_\varepsilon(\widehat{K}_{ij}, u^{(i)}, v^{(j)})$.
- Apply mini-batch randomized block coordinate descent (MRBCD).

for An agent pair $(i, j) \sim E$ **do**

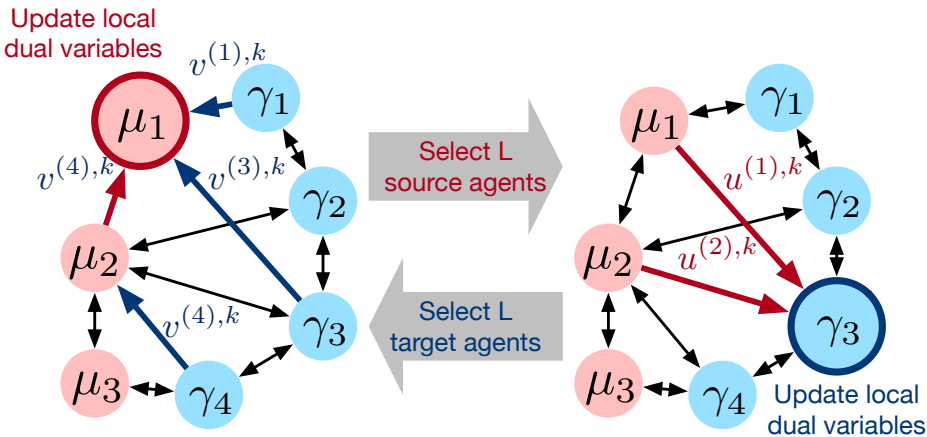
Select L target agents $\mathcal{J}_L \sim \frac{1}{\|E[i,:]\|_1} E[i,:]$. Send $\{v^{(j),t}\}_{j \in \mathcal{J}_L}$ to the source agent i

$u^{(i),t+1} \leftarrow u^{(i),t} + \eta_t \sum_{j \in \mathcal{J}_L} \nabla_{u^{(i)}} \hat{f}_\varepsilon^{(i,j),t}$

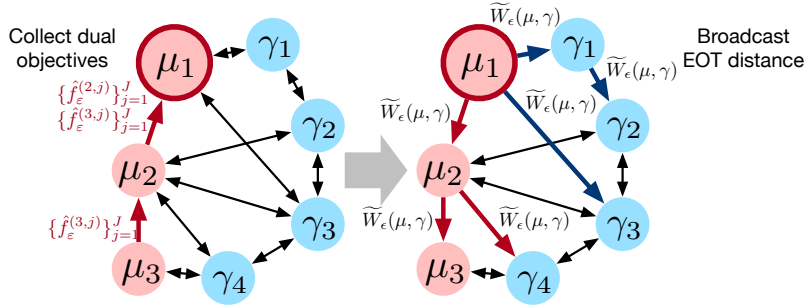
Select L source agents $\mathcal{I}_L \sim \frac{1}{\|E[:,j]\|_1} E[:,j]$. Send $\{u^{(i),t}\}_{i \in \mathcal{I}_L}$ to the target agent j

$v^{(j),t+1} \leftarrow v^{(j),t} + \eta_t \sum_{i \in \mathcal{I}_L} \nabla_{v^{(j)}} \hat{f}_\varepsilon^{(i,j),t}$

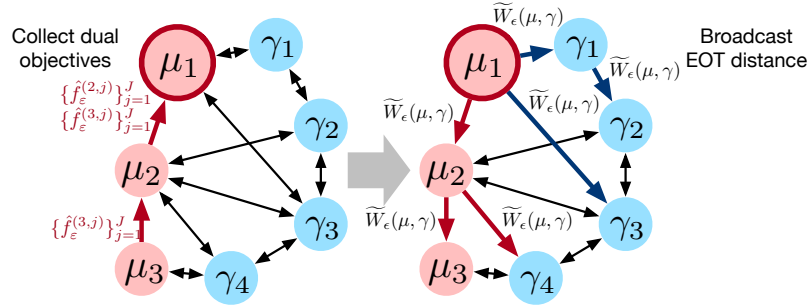
Step 2 of DEOT: Update Dual Variables via MRBCD



Step 3 of DEOT: Compute and Broadcast EOT Distance



Step 3 of DEOT: Compute and Broadcast EOT Distance



Lemma (Convergence Analysis, based on (Wang, et al., 2014))

Let $\|\nabla_{u,v} F_\varepsilon\|_2 \le R$, $R_0 = \min_{u,v\in\mathcal{C}^*} \|(u^0, v^0) - (u, v)\|_2$, *and* F_ε *be* L_{F_ε}*-Lipschitz:*

$$\mathbb{E}|F_\varepsilon(\hat{u}^t, \hat{v}^t; \widehat{K}, E) - F_\varepsilon(u^*, v^*; \widehat{K}, E)| \le \mathcal{O}\Big(\frac{IJ\left((\sqrt{t} + L_{F_\varepsilon})R_0^2 + \sqrt{t}R^2\right)}{t}\Big). \quad (105)$$

Theoretical Analysis of Approximation Error

Let $F_\varepsilon(u, v; K, E)$ be the objective with specific dual variables u, v and under a kernel K and a communication protocol E.

$$\underbrace{\mathbb{E}\left[\left|F_\varepsilon(\hat{u}^t, \hat{v}^t; \widehat{K}, E) - W_\varepsilon(\mu, \gamma)\right|\right]}_{\text{Approximation Error}}$$

Theoretical Analysis of Approximation Error

Let $F_\varepsilon(u, v; K, E)$ be the objective with specific dual variables u, v and under a kernel K and a communication protocol E.

$$\underbrace{\mathbb{E}\left[\left|F_\varepsilon(\hat{u}^t, \hat{v}^t; \widehat{K}, E) - W_\varepsilon(\mu, \gamma)\right|\right]}_{\text{Approximation Error}} = \mathbb{E}\Big[\underbrace{\left|F_\varepsilon(\hat{u}^t, \hat{v}^t; \widehat{K}, E) - F_\varepsilon(\hat{u}^*, \hat{v}^*; \widehat{K}, E)\right|}_{\text{Convergence Error by Lemma 1}}\Big] + \underbrace{\left|F_\varepsilon(\hat{u}^*, \hat{v}^*; \widehat{K}, E) - F_\varepsilon(\tilde{u}^*, \tilde{v}^*; K, E)\right|}_{\text{Kernel Error based on Theorem 2}} + \underbrace{\overbrace{\left|\widetilde{W}_\varepsilon(\mu,\gamma) - W_\varepsilon(\mu,\gamma)\right|}^{\text{Gap by Theorem 1}} + \left|F_\varepsilon(\tilde{u}^*, \tilde{v}^*; K, E) - W_\varepsilon(\mu, \gamma)\right|}} \tag{106}$$

Theoretical Analysis of Approximation Error

Quantify the influence of kernel approximation error on optimal objective.

- F_ε is Lipschitz continuous with respect to (u, v) [Genevay, et al. NeurIPS 2016].

Theoretical Analysis of Approximation Error

Quantify the influence of kernel approximation error on optimal objective.

- F_ε is Lipschitz continuous with respect to (u, v) [Genevay, et al. NeurIPS 2016].
- F_ε is a linear and Lipschitz continuous function with respect to K.

Theoretical Analysis of Approximation Error

Quantify the influence of kernel approximation error on optimal objective.

- F_ε is Lipschitz continuous with respect to (u, v) [Genevay, et al. NeurIPS 2016].
- F_ε is a linear and Lipschitz continuous function with respect to K.

Lemma ([Dempe, et al. JGO 2015)

Lemma 3.1] Let $\phi(K) = \max_{u,v} F_\varepsilon\big(u, v; K, E\big)$ *be the optimal objective function with respect to* K*. It is* L_κ*-Lipschitz continuous:*

$$\big|F_\varepsilon(\hat{u}^*, \hat{v}^*; \widehat{K}, E) - F_\varepsilon(\tilde{u}^*, \tilde{v}^*; K, E)\big| = \big|\phi(\widehat{K}) - \phi(K)\big| \leq L_\kappa \big\|\widehat{K} - K\big\|. \qquad (107)$$

Theoretical Analysis of Approximation Error

Quantify the influence of kernel approximation error on optimal objective.

- F_ε is Lipschitz continuous with respect to (u, v) [Genevay, et al. NeurIPS 2016].
- F_ε is a linear and Lipschitz continuous function with respect to K.

Lemma ([Dempe, et al. JGO 2015)

Lemma 3.1] Let $\phi(K) = \max_{u,v} F_\varepsilon\big(u, v; K, E\big)$ *be the optimal objective function with respect to* K. *It is* L_κ*-Lipschitz continuous:*

$$\big|F_\varepsilon(\hat{u}^*, \hat{v}^*; \widehat{K}, E) - F_\varepsilon(\tilde{u}^*, \tilde{v}^*; K, E)\big| = \big|\phi(\widehat{K}) - \phi(K)\big| \leq L_\kappa \big\|\widehat{K} - K\big\|. \qquad (107)$$

Theorem (Error Bound)

With probability at least $1 - \delta$, *we have*

$$\mathbb{E}|F_\varepsilon(\hat{u}^t, \hat{v}^t; \widehat{K}, E) - W_\varepsilon(\mu, \gamma)| \leq \mathcal{O}\Big(\frac{IJ}{\sqrt{t}} + (N + M)\sqrt{\frac{1}{P}\log\frac{2(N + M)}{\delta}} + \sigma\Big). \qquad (108)$$

Experiments on Synthetic Data

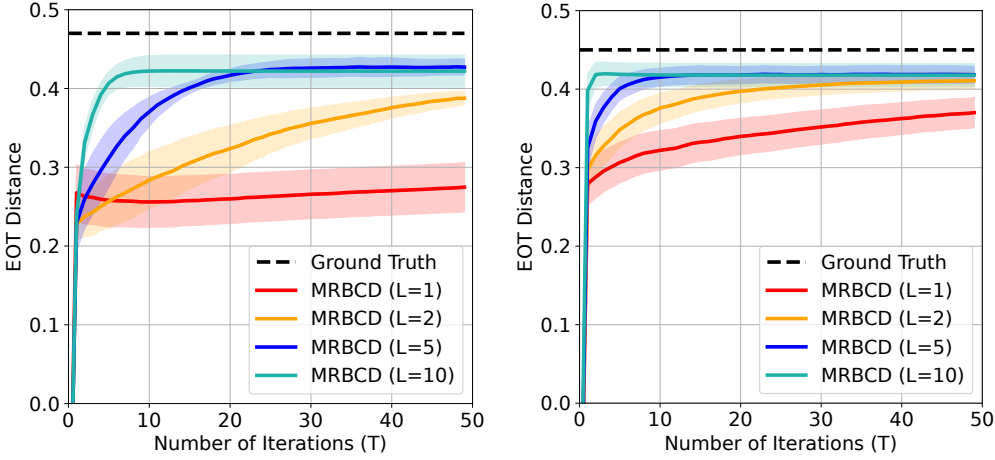
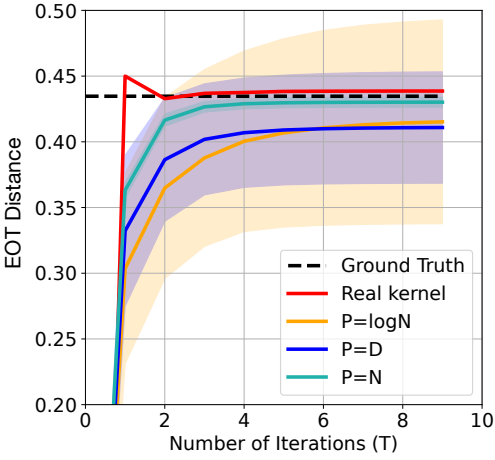
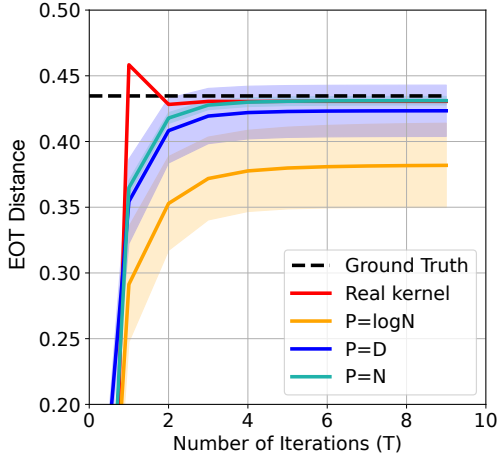


Figure: Empirical convergence under different batch sizes

Experiments on Synthetic Data



(a) $D = 10$



(b) $D = 50$

Figure: Influences of kernel approximation precision

Experiments on Synthetic Data

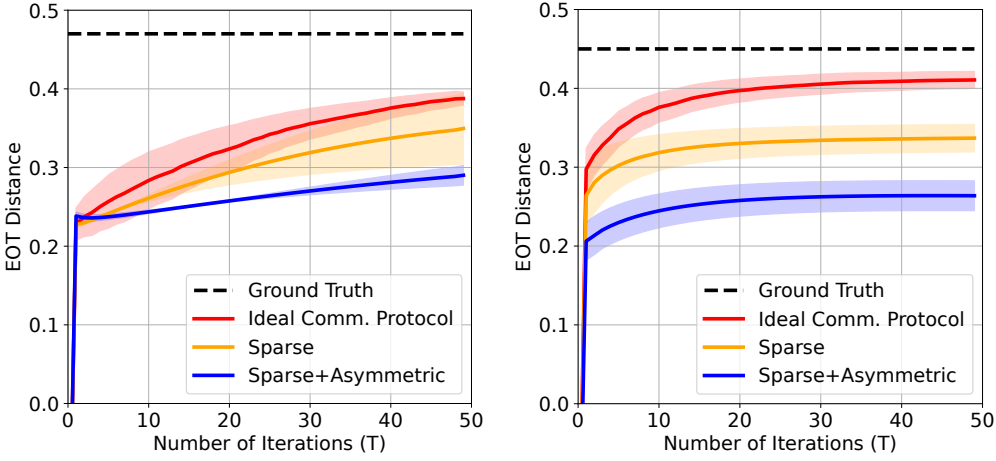


Figure: Influences of various communication protocols

Experiments on Synthetic Data

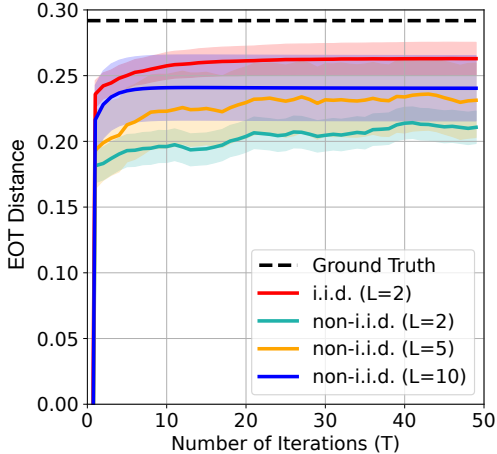


Figure: Influences of various storage protocols

Experiments on (Distributed) Domain Adaptation

Structure	Method	USPS → MNIST	MNIST → USPS	Preserve Privacy
Source only	1NN	0.385	0.593	Yes
Centralized	EMD	0.544	0.617	No
	Sinkhorn	0.437	0.620	No
	OT-LpL1	0.490	0.676	No
Decentralized (Ours)	MRBCD$_K$	0.580	0.681	No
	MRBCD$_{\widehat{K}}$	0.522	0.629	Yes

Experiments on (Distributed) Domain Adaptation

Domains	Source only	Centralized			Decentralized (Ours)	
	1NN	EMD	Sinkhorn	OT-LpL1	MRBCD_K	$\text{MRBCD}_{\widehat{K}}$
Ar→Cl	0.433	0.471	0.492	0.490	0.483	0.458
Ar→Pr	0.594	0.642	0.673	0.633	0.665	0.639
Ar→Rw	0.667	0.677	0.721	0.686	0.738	0.705
Cl→Ar	0.445	0.504	0.509	0.478	0.531	0.509
Cl→Pr	0.536	0.647	0.617	0.642	0.632	0.606
Cl→Rw	0.589	0.638	0.657	0.664	0.654	0.618
Pr→Ar	0.488	0.516	0.532	0.494	0.538	0.506
Pr→Cl	0.414	0.455	0.465	0.450	0.469	0.425
Pr→Rw	0.683	0.707	0.725	0.714	0.735	0.704
Rw→Ar	0.592	0.611	0.622	0.605	0.621	0.598
Rw→Cl	0.450	0.498	0.505	0.509	0.494	0.463
Rw→Pr	0.729	0.749	0.778	0.770	0.773	0.736

Summary

- Optimal transport leads to new learning paradigms for unaligned data, and thus is helpful for both privacy protection and model robustness.
  - The correspondence among samples can be inferred by OT plans
  - The aggregation/fusion of information can be solved as a barycenter problem
  - Achieve encouraging performance on robust multi-modal learning

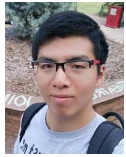
Summary

- Optimal transport leads to new learning paradigms for unaligned data, and thus is helpful for both privacy protection and model robustness.
  - The correspondence among samples can be inferred by OT plans
  - The aggregation/fusion of information can be solved as a barycenter problem
  - Achieve encouraging performance on robust multi-modal learning
- It is possible to solve optimal transport problem approximately in a decentralized way, with data sharing.
  - The dual-forms of OT problems are friendly for distributed learning
  - The approximation error is determined by multiple factors and can be analyzed quantitatively
  - Achieve encouraging performance on (distributed) domain adaptation.

Acknowledgment

Computational Optimal Transport


Xiangfeng Wang
ECNU


Cheng Meng
RUC


Jun Yu
BIT


Tao Li
RUC

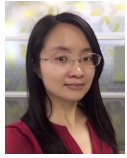

Mengyu Li
RUC

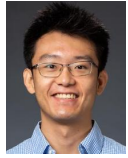

Moyi Yang
ECNU

OT-based Machine Learning


Lawrence Carin
Duke


Hongyuan Zha
CUHK-SZ


Dixin Luo
BIT


Jiachang Liu
Duke


Minjie Cheng
RUC


Fengjiao Gong
RUC


Yuzhou Nie
RUC

Thank you!

https://hongtengxu.github.io

https://github.com/HongtengXu

hongtengxu@ruc.edu.cn

AAAI'22 Tutorial on Gromov-Wasserstein Learning
https://hongtengxu.github.io/talks.html