Recent Works on Optimal Transport

Talgat Daulbaev
Anna Petrovskaya
Ekaterina Sosnina

April 11, 2019

Table of contents

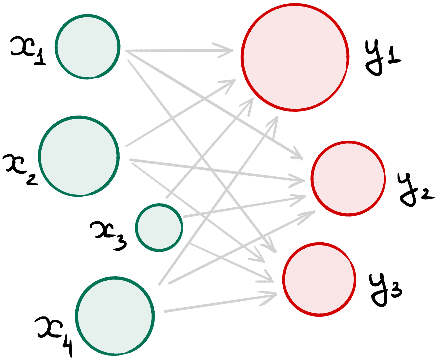
Recapitulation

Discrete-Discrete Optimal Transport

- Point clouds:

$$\{\boldsymbol{x}_i\}_{i=1}^m \text{ and } \{\boldsymbol{y}_i\}_{i=1}^n$$

- Let r_i be the mass of $\boldsymbol{x}_i$, and c_j be the desired mass of $\boldsymbol{y}_j$
- $\boldsymbol{M}_{ij}$ is the cost of moving a unit mass from $\boldsymbol{x}_i$ to $\boldsymbol{y}_j$ (ground cost matrix)
- We want to transport masses in an optimal way



Optimal Transport

Let $\boldsymbol{r} \in \boldsymbol{\Sigma}_m$ and $\boldsymbol{c} \in \boldsymbol{\Sigma}_n$ be the vectors which correspond to the mass of each point, where $\boldsymbol{\Sigma}_d \triangleq \{\boldsymbol{x} \in \mathbb{R}_+^d : \; \boldsymbol{x}^\top \mathbb{1}_d = 1\}$

Each transportation plan is defined by a matrix from

$$\boldsymbol{U}(\boldsymbol{r}, \boldsymbol{c}) \triangleq \{\boldsymbol{P} \in \mathbb{R}_+^{m \times n} : \; \boldsymbol{P}\mathbb{1}_n = \boldsymbol{r}, \boldsymbol{P}^\top \mathbb{1}_m = \boldsymbol{c}\}$$

$\boldsymbol{M}_{ij}$ is the cost of moving a unit mass from $\boldsymbol{x}_i$ to $\boldsymbol{y}_j$

Optimal Transport distance:

$$d_{\boldsymbol{M}}(\boldsymbol{r}, \boldsymbol{c}) \triangleq \min_{\boldsymbol{P} \in \boldsymbol{U}(\boldsymbol{r}, \boldsymbol{c})} \langle \boldsymbol{P}, \boldsymbol{M} \rangle$$

Drawbacks: Ill-Posedness

$$d_{\boldsymbol{M}}(\boldsymbol{r}, \boldsymbol{c}) \triangleq \min_{\boldsymbol{P} \in \boldsymbol{U}(\boldsymbol{r}, \boldsymbol{c})} \langle \boldsymbol{P}, \boldsymbol{M} \rangle$$

✓ a solution exists

× the solution is not unique

× the solution is “unstable”

Drawbacks: Complexity

Definition

$f(n) \in \widetilde{\mathcal{O}}(h(n))$, when $\exists k \in \mathbb{N}$: $f(n) \in \mathcal{O}(h(n) \log^k h(n))$

Optimal Transport Complexity

Best theoretical: $\widetilde{\mathcal{O}}(n^{2.5})$ [Lee & Sidford, 2014]

Best practical: $\widetilde{\mathcal{O}}(n^3)$ [e.g., min cost flow solver]

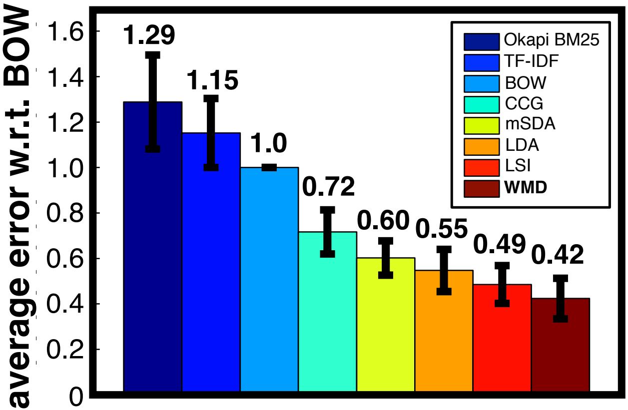
Example: Word Mover's Distance

OT distance is a perfect tool to define a distance between sets, consisting of metric space objects

$\boldsymbol{r}$ and $\boldsymbol{c}$ are term-frequency vectors, $\boldsymbol{M}_{ij}$ is the euclidean distance between word2vec representations of x_i and y_j

"From Word Embeddings To Document Distances" by Kusner et al. (ICML 2015)

Word Mover's Distance: Results



The kNN test errors of various document metrics averaged over eight datasets, relative to kNN with BOW

Wasserstein on Discrete Measures

If the ground cost matrix $\boldsymbol{M}$ is defined by a distance (e.g., $\boldsymbol{M}_{ij} \triangleq \|\boldsymbol{x}_i - \boldsymbol{y}_j\|_1$), the optimal transport distance is called the Wasserstein distance

It is a special case of the Wasserstein distance between two discrete measures $\mu = \sum_{i=1}^{n} a_i \delta_{\boldsymbol{x}_i}$ and $\nu = \sum_{i=1}^{n} b_i \delta_{\boldsymbol{y}_i}$

$$W_p(\mu, \nu) := \left(\inf_{\gamma \in \Gamma(\mu,\nu)} \int_{M \times M} d(x, y)^p \, \mathrm{d}\gamma(x, y) \right)^{1/p}$$

Slide that should remind you what's going on

Let $\boldsymbol{r} \in \boldsymbol{\Sigma}_m$ and $\boldsymbol{c} \in \boldsymbol{\Sigma}_n$ be the vectors which correspond to the mass of each point, where $\boldsymbol{\Sigma}_d \triangleq \{\boldsymbol{x} \in \mathbb{R}_+^d : \ \boldsymbol{x}^\top \mathbb{1}_d = 1\}$

Each transportation plan is defined by a matrix from

$$\boldsymbol{U}(\boldsymbol{r}, \boldsymbol{c}) \triangleq \{\boldsymbol{P} \in \mathbb{R}_+^{m \times n} : \ \boldsymbol{P}\mathbb{1}_n = \boldsymbol{r}, \boldsymbol{P}^\top \mathbb{1}_m = \boldsymbol{c}\}$$

$\boldsymbol{M}_{ij}$ is the cost of moving a unit mass from $\boldsymbol{x}_i$ to $\boldsymbol{y}_j$

Optimal Transport distance:

$$d_{\boldsymbol{M}}(\boldsymbol{r}, \boldsymbol{c}) \triangleq \min_{\boldsymbol{P} \in \boldsymbol{U}(\boldsymbol{r}, \boldsymbol{c})} \langle \boldsymbol{P}, \boldsymbol{M} \rangle$$

“Sinkhorn Distances: Lightspeed Computation of Optimal Transport” by Cuturi (NIPS 2013)

Entropy Regularization

Sinkhorn (dual) distance:

$$d_{\boldsymbol{M}}^{\lambda}(\boldsymbol{r}, \boldsymbol{c}) \triangleq \min_{\boldsymbol{P} \in \boldsymbol{U}(\boldsymbol{r}, \boldsymbol{c})} \langle \boldsymbol{P}, \boldsymbol{M} \rangle - \frac{1}{\lambda} h(\boldsymbol{P}),$$

where $h(\boldsymbol{P})$ is the entropy

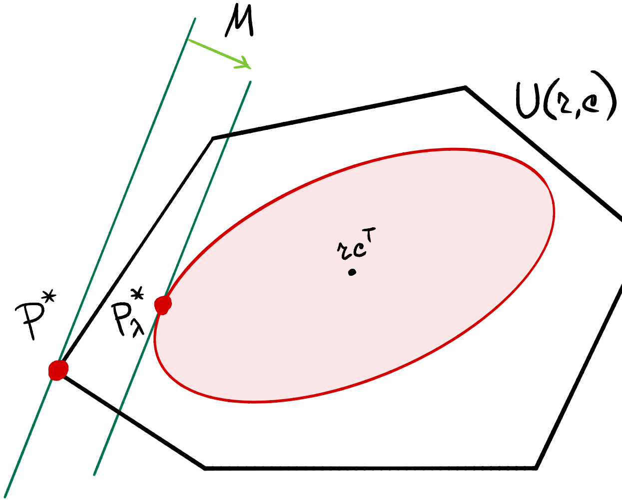
$$h(\boldsymbol{P}) \triangleq - \sum_{ij} \boldsymbol{P}_{ij} \log \boldsymbol{P}_{ij}$$

$$\boldsymbol{U}(\boldsymbol{r}, \boldsymbol{c}) \triangleq \{ \boldsymbol{P} \in \mathbb{R}_{+}^{m \times n} : \ \boldsymbol{P} \mathbb{1}_n = \boldsymbol{r}, \boldsymbol{P}^{\top} \mathbb{1}_m = \boldsymbol{c} \}$$

It can be shown that there exists $\alpha > 0$, such that

$$d_{\boldsymbol{M}}^{\lambda}(\boldsymbol{r}, \boldsymbol{c}) = \min_{\boldsymbol{P} \in \boldsymbol{U}(\boldsymbol{r}, \boldsymbol{c}),\ KL(\boldsymbol{P} \| \boldsymbol{r} \boldsymbol{c}^{\top}) < \alpha} \langle \boldsymbol{P}, \boldsymbol{M} \rangle$$

Geometric Interpretation



Sinkhorn-Knopp Lemma

Lemma

For $\lambda > 0$, $\boldsymbol{P}^{\lambda} = \mathrm{diag}(\boldsymbol{u})\boldsymbol{K}\mathrm{diag}(\boldsymbol{v})$, *where* $\boldsymbol{u} \in \mathbb{R}_{+}^{m}$ *and* $\boldsymbol{v} \in \mathbb{R}_{+}^{n}$ *are uniquely defined up to a multiplicative factor and* $\boldsymbol{K} \triangleq e^{-\lambda \boldsymbol{M}}$ *is the element-wise exponential of* $-\lambda \boldsymbol{M}$.

Proof.

$$\mathcal{L}(\boldsymbol{P}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \sum_{i,j} \left\{ \boldsymbol{P}_{ij} \boldsymbol{M}_{ij} + \frac{1}{\lambda} \boldsymbol{P}_{ij} \log \boldsymbol{P}_{ij} \right\} + \boldsymbol{\alpha}^{\top} (\boldsymbol{P} \mathbb{1}_n - \boldsymbol{r}) + \boldsymbol{\beta}^{\top} \left(\boldsymbol{P}^{\top} \mathbb{1}_m - \boldsymbol{c} \right)$$

$$\frac{\partial \mathcal{L}(\boldsymbol{P}, \boldsymbol{\alpha}, \boldsymbol{\beta})}{\partial \boldsymbol{P}_{ij}} = \boldsymbol{M}_{ij} + \frac{1}{\lambda} \log \boldsymbol{P}_{ij} + 1 + \boldsymbol{\alpha}_i + \boldsymbol{\beta}_j = 0$$

$$\log \boldsymbol{P}_{ij} = (-\boldsymbol{\alpha}_i - 1/2) + (-\lambda \boldsymbol{M}_{ij}) + (-\boldsymbol{\beta}_j - 1/2)$$

□

Sinkhorn – Knopp Algorithm

How to find a non-negative matrix $\boldsymbol{P}$, such that

$$\boldsymbol{P} = \text{diag}(\boldsymbol{u})\boldsymbol{K}\text{diag}(\boldsymbol{v}) : \quad \sum_{i=1}^{m} \boldsymbol{P}_{ij} = c_j \text{ and } \sum_{i=j}^{n} \boldsymbol{P}_{ij} = r_i?$$

Take arbitrary non-negative vectors $\boldsymbol{u}^{(0)}$ and $\boldsymbol{v}^{(0)}$ scale the matrix $\boldsymbol{P}$ until convergence

Sinkhorn – Knopp Algorithm

$$\boxed{\boldsymbol{P} = \mathrm{diag}(\boldsymbol{u})\boldsymbol{K}\mathrm{diag}(\boldsymbol{v}) : \quad \sum_{i=1}^{m} \boldsymbol{P}_{ij} = c_j \text{ and } \sum_{i=j}^{n} \boldsymbol{P}_{ij} = r_i}$$

$$\boldsymbol{P}_{ij} = u_i \boldsymbol{K}_{ij} v_j$$

$$\sum_j \boldsymbol{P}_{ij} = u_i \sum_j \boldsymbol{K}_{ij} u_j \longleftrightarrow r_i \qquad u_i = r_i \,/\, (\boldsymbol{K}\boldsymbol{v})_i$$

$$\sum_i \boldsymbol{P}_{ij} = v_j \sum_j \boldsymbol{K}_{ij} u_i \longleftrightarrow c_j \qquad v_j = c_j \,/\, \left(\boldsymbol{K}^\top \boldsymbol{u}\right)_j$$

$$\left[\begin{array}{l} \boldsymbol{u}^{(k+1)} \leftarrow \boldsymbol{r} \,/\, \left(\boldsymbol{K}\boldsymbol{v}^{(k)}\right) \\ \boldsymbol{v}^{(k+1)} \leftarrow \boldsymbol{c} \,/\, \left(\boldsymbol{K}^\top \boldsymbol{u}^{(k+1)}\right) \end{array}\right.$$

“Near-linear time approximation algorithms for optimal transport via Sinkhorn iteration” by Altschuler et al. (NIPS 2017)

Task Formulation and Contributions

The goal is to find an approximate optimal plan $\widehat{\boldsymbol{P}} \in \boldsymbol{U}(\boldsymbol{r}, \boldsymbol{c})$ satisfying

$$\langle \widehat{\boldsymbol{P}}, \boldsymbol{M} \rangle \leq \min_{\boldsymbol{P} \in \boldsymbol{U}(\boldsymbol{r}, \boldsymbol{v})} \langle \boldsymbol{P}, \boldsymbol{M} \rangle + \varepsilon$$

Two major contributions:

- The Sinkhorn-Knopp algorithm's complexity is proven to be $\mathcal{O}\left(n^2 \varepsilon^{-3} \log(n) \|\boldsymbol{M}\|_\infty^3\right)$
- Greenkhorn: a new greedy algorithm for computing Sinkhorn distance with the same theoretical complexity

Sketches of the Algorithms

Sinkhorn-Knopp (1967)

1. Compute $\boldsymbol{K} \leftarrow e^{-\lambda \boldsymbol{M}}$
2. Alternately rescale rows/columns to match $\boldsymbol{r}$ and $\boldsymbol{c}$

Greenkhorn (2017)

1. Compute $\boldsymbol{K} \leftarrow e^{-\lambda \boldsymbol{M}}$
2. Greedily rescale one row/column to match $\boldsymbol{r}$ and $\boldsymbol{c}$

Approximate Optimal Transport Algorithms

Algorithm 1 $\textsc{ApproxOT}(C,\ r,\ c,\ \varepsilon)$

$\eta \leftarrow \frac{4\log n}{\varepsilon}$, $\varepsilon' \leftarrow \frac{\varepsilon}{8\|C\|_\infty}$

\\\\ Step 1: Approximately project onto $\mathcal{U}_{r,c}$

$A \leftarrow \exp(-\eta C)$

$B \leftarrow \textsc{Proj}(A, \mathcal{U}_{r,c}, \varepsilon')$

\\\\ Step 2: Round to feasible point in $\mathcal{U}_{r,c}$

Output $\hat{P} \leftarrow \textsc{round}(B, \mathcal{U}_{r,c})$

Algorithm 2 $\textsc{round}(F, \mathcal{U}_{r,c})$

$X \leftarrow \mathbf{D}(x)$ with $x_i = \frac{r_i}{r_i(F)} \wedge 1$

$F' \leftarrow XF$

$Y \leftarrow \mathbf{D}(y)$ with $y_j = \frac{c_j}{c_j(F')} \wedge 1$

$F'' \leftarrow F'Y$

$\mathrm{err}_r \leftarrow r - r(F'')$, $\mathrm{err}_c \leftarrow c - c(F'')$

Output $G \leftarrow F'' + \mathrm{err}_r \mathrm{err}_c^\top / \|\mathrm{err}_r\|_1$

Algorithms

Algorithm 3 $\textsc{Sinkhorn}(A, \mathcal{U}_{r,c}, \varepsilon')$

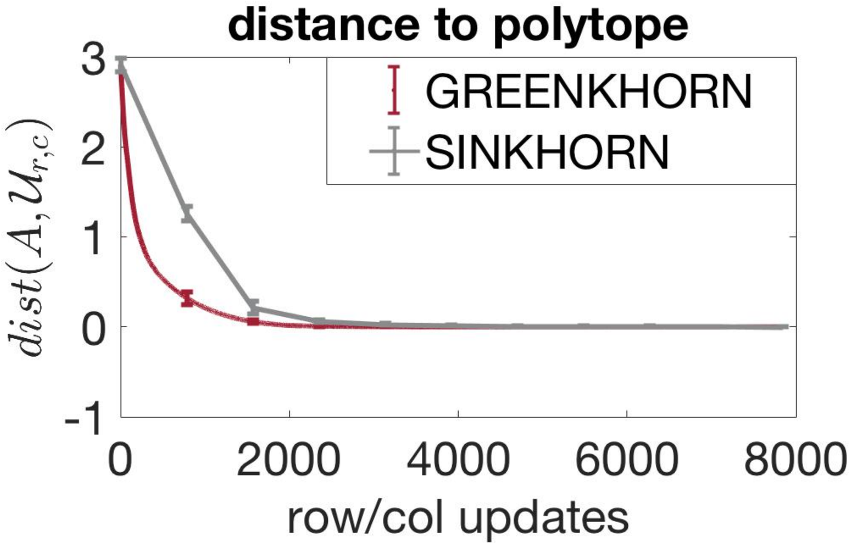
Initialize $k \leftarrow 0$
$A^{(0)} \leftarrow A/\|A\|_1$, $x^0 \leftarrow \mathbf{0}$, $y^0 \leftarrow \mathbf{0}$
while $\mathrm{dist}(A^{(k)}, \mathcal{U}_{r,c}) > \varepsilon'$ **do**
    $k \leftarrow k + 1$
    if k odd **then**
        $x_i \leftarrow \log \frac{r_i}{r_i(A^{(k-1)})}$ for $i \in [n]$
        $x^k \leftarrow x^{k-1} + x$, $y^k \leftarrow y^{k-1}$
    else
        $y \leftarrow \log \frac{c_j}{c_j(A^{(k-1)})}$ for $j \in [n]$
        $y^k \leftarrow y^{k-1} + y$, $x^k \leftarrow x^{k-1}$
    $A^{(k)} = \mathbf{D}(\exp(x^k)) A \mathbf{D}(\exp(y^k))$
Output $B \leftarrow A^{(k)}$

Algorithm 4 $\textsc{Greenkhorn}(A, \mathcal{U}_{r,c}, \varepsilon')$

$A^{(0)} \leftarrow A/\|A\|_1$, $x \leftarrow \mathbf{0}$, $y \leftarrow \mathbf{0}$.
$A \leftarrow A^{(0)}$
while $\mathrm{dist}(A, \mathcal{U}_{r,c}) > \varepsilon$ **do**
    $I \leftarrow \mathrm{argmax}_i\, \rho(r_i, r_i(A))$
    $J \leftarrow \mathrm{argmax}_j\, \rho(c_j, c_j(A))$
    if $\rho(r_I, r_I(A)) > \rho(c_J, c_J(A))$ **then**
        $x_I \leftarrow x_I + \log \frac{r_I}{r_I(A)}$
    else
        $y_J \leftarrow y_J + \log \frac{c_J}{c_J(A)}$
    $A \leftarrow \mathbf{D}(\exp(x)) A^{(0)} \mathbf{D}(\exp(y))$
Output $B \leftarrow A$

$$\mathrm{dist}\left(\boldsymbol{A}, \mathcal{U}_{\boldsymbol{r},\boldsymbol{c}}\right) = \|\boldsymbol{r}(\boldsymbol{A}) - \boldsymbol{r}\|_1 + \|c(\boldsymbol{A}) - \boldsymbol{c}\|_1, \quad \rho(a, b) = b - a + a \log \frac{a}{b}$$

Empirical Convergence



“Smooth and Sparse Optimal Transport” by Blondel et al. (AISTATS 2018)

Entropy is not the only Regularization

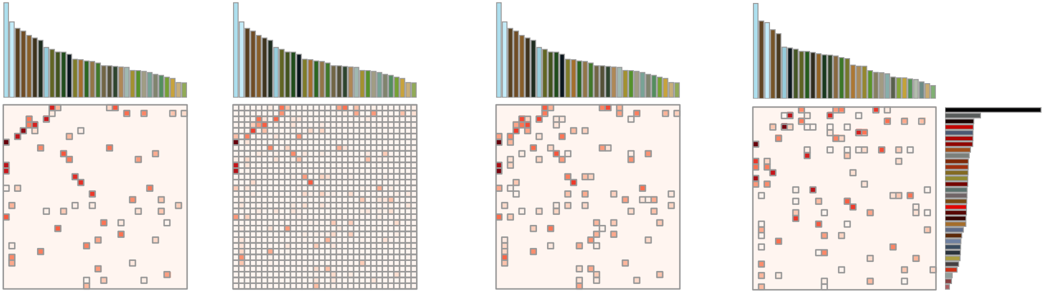
We can take any **strongly** convex function $\mathcal{R}$ and define a regularized optimal transport as

$$\widehat{d}_{\boldsymbol{M}}(\boldsymbol{r}, \boldsymbol{c}) \triangleq \min_{\boldsymbol{P} \in \boldsymbol{U}(\boldsymbol{r}, \boldsymbol{c})} \{\langle \boldsymbol{P}, \boldsymbol{M} \rangle + \gamma \mathcal{R}(\boldsymbol{P})\}$$

Smooth and Sparse Optimal Plans

Wasserstein optimal plans are often sparse, but Sinkhorn transportation matrices are **not** sparse

Why? Because at least $\log(0)$ is not defined



Unregularized
Sparsity: 94%

Smoothed semi-dual (ent.)
Sparsity: 0%

Smoothed semi-dual (sq. 2-norm)
Sparsity: 90%

Semi-relaxed primal (Eucl.)
Sparsity: 91%

Dual and Semi-Dual Problems

Dual:

$$\mathrm{OT}(\boldsymbol{r}, \boldsymbol{c}) = \max_{\boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathcal{P}(M)} \boldsymbol{\alpha}^\top \boldsymbol{r} + \boldsymbol{\beta}^\top \boldsymbol{c},$$

$$\mathcal{P}(M) := \{\boldsymbol{\alpha} \in \mathbb{R}^m, \boldsymbol{\beta} \in \mathbb{R}^n : \alpha_i + \beta_j \leq M_{i,j}\}$$

If $\boldsymbol{\alpha}$ is fixed, an optimal solution w.r.t. $\boldsymbol{\beta}$ is

$$\beta_j = \min_{i \in \{1,\ldots,m\}} M_{i,j} - \alpha_i, \quad \forall j \in \{1, \ldots, n\}$$

Semi-Dual:

$$\mathrm{OT}(\boldsymbol{r}, \boldsymbol{c}) = \max_{\boldsymbol{\alpha} \in \mathbb{R}^m} \boldsymbol{\alpha}^\top \boldsymbol{r} - \sum_{j=1}^{n} c_j \max_{i \in \{1,\ldots,m\}} (\alpha_i - M_{i,j})$$

Smooth Relaxed Dual

Indicator:

$$\delta(\boldsymbol{x}) \triangleq \begin{cases} 0, & \text{if } \boldsymbol{x} \leq 0 \\ \infty, & \text{otherwise} \end{cases} \qquad = \sup_{\boldsymbol{y} \geq 0} \boldsymbol{y}^\top \boldsymbol{x}$$

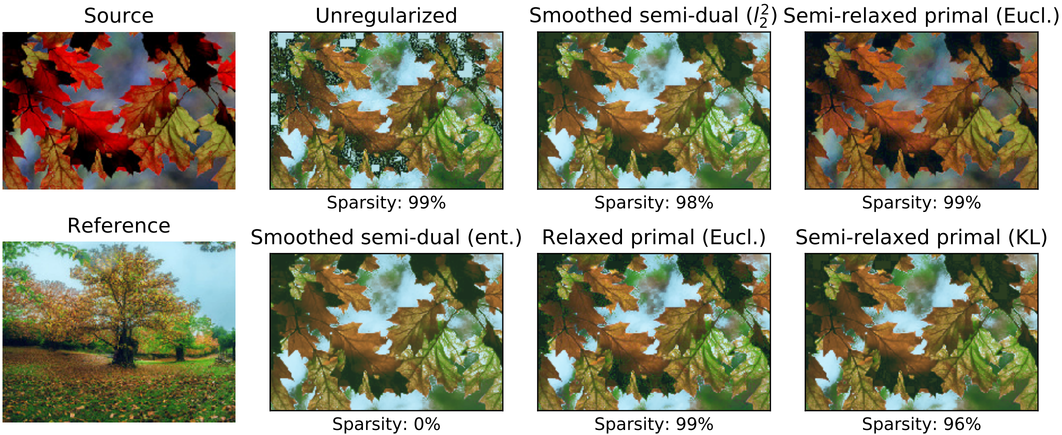
Smoothed version of δ:

$$\delta_\Omega(\boldsymbol{x}) \triangleq \sup_{\boldsymbol{y} \geq 0} \boldsymbol{y}^\top \boldsymbol{x} - \Omega(\boldsymbol{y})$$

Smoothed relaxed dual:

$$\mathrm{OT}_\Omega(\boldsymbol{r}, \boldsymbol{c}) = \max_{\substack{\boldsymbol{\alpha} \in \mathbb{R}^m \\ \boldsymbol{\beta} \in \mathbb{R}^n}} \boldsymbol{\alpha}^\top \boldsymbol{r} + \boldsymbol{\beta}^\top \boldsymbol{c} - \sum_{j=1}^{n} \delta_\Omega\left(\boldsymbol{\alpha} + \beta_j \mathbf{1}_m - \boldsymbol{M}_j\right)$$

Results


Source
Unregularized
Smoothed semi-dual (l_2^2)
Semi-relaxed primal (Eucl.)
Sparsity: 99%
Sparsity: 98%
Sparsity: 99%
Reference
Smoothed semi-dual (ent.)
Relaxed primal (Eucl.)
Semi-relaxed primal (KL)
Sparsity: 0%
Sparsity: 99%
Sparsity: 96%

“Computational Optimal Transport: Complexity by Accelerated Gradient Descent Is Better Than by Sinkhorn’s Algorithm” by Dvurechensky et al. (ICML 2018)

Contributions

- Improved complexity for approximating the OT distance:

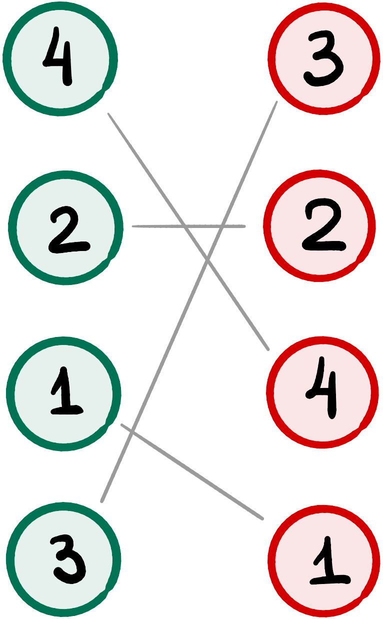
$$\mathcal{O}\left(\frac{n^2\|M\|_\infty^2 \ln n}{\varepsilon^2}\right)$$

- An Adaptive Primal-Dual Accelerated Gradient Descent (APDAGD) algorithm: a flexible framework for OT problems with different regularizers
- Improved complexity for approximating the OT distance, by APDAGD method

$$\mathcal{O}\left(\min\left\{\frac{n^{9/4}\sqrt{\|C\|_\infty R \ln n}}{\varepsilon}, \frac{n^2\|M\|_\infty R \ln n}{\varepsilon^2}\right\}\right)$$

“Learning Latent Permutations with Gumbel-Sinkhorn Networks” by Mena et al. (ICLR 2018)

Learning Permutations



Permutations as a Transportation Plan

$$\pi : \{1, \ldots, m\} \to \{1, \ldots, m\}$$

$$P_\pi = \begin{bmatrix} \mathbf{e}_{\pi(1)} \\ \mathbf{e}_{\pi(2)} \\ \mathbf{e}_{\pi(3)} \\ \mathbf{e}_{\pi(4)} \\ \mathbf{e}_{\pi(5)} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \qquad P_\pi \mathbf{g} = \begin{bmatrix} \mathbf{e}_{\pi(1)} \\ \mathbf{e}_{\pi(2)} \\ \vdots \\ \mathbf{e}_{\pi(n)} \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ \vdots \\ n \end{bmatrix} = \begin{bmatrix} \pi(1) \\ \pi(2) \\ \vdots \\ \pi(n) \end{bmatrix}$$

Matching operator gives mapping from unconstrained matrices to permutations:

$$M(X) = \arg\max_{P \in \mathcal{P}_N} \langle P, X \rangle_F,$$

where $\mathcal{P}_N$ is a set of all permutation matrices

Credits: https://duvenaud.github.io/

Relaxing Permutations

Birkhoff Polytope: $\mathcal{B}_N = \left\{ A \in \mathbb{R}^{N\times N} | \sum_i a_{ij} = \sum_j a_{ij} = 1 \right\}$

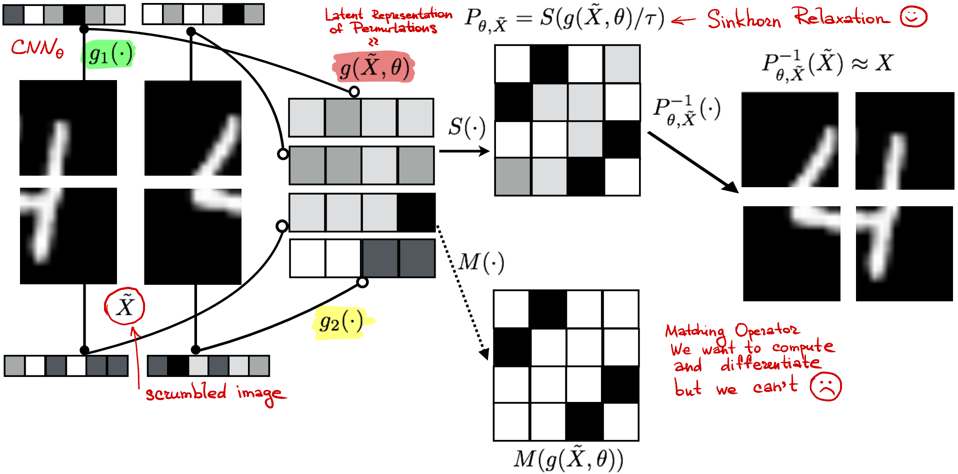
Sinkhorn Operator: $S(\Phi/\tau) = \arg\max_{P\in\mathcal{B}_N} \langle P, \Phi \rangle_F + \tau h(P)$

Theorem

If the entries of X are drawn independently from a distribution that is absolutely continuous with respect to the Lebesgue measure in $\mathbb{R}$. Then, almost surely, the following convergence holds:

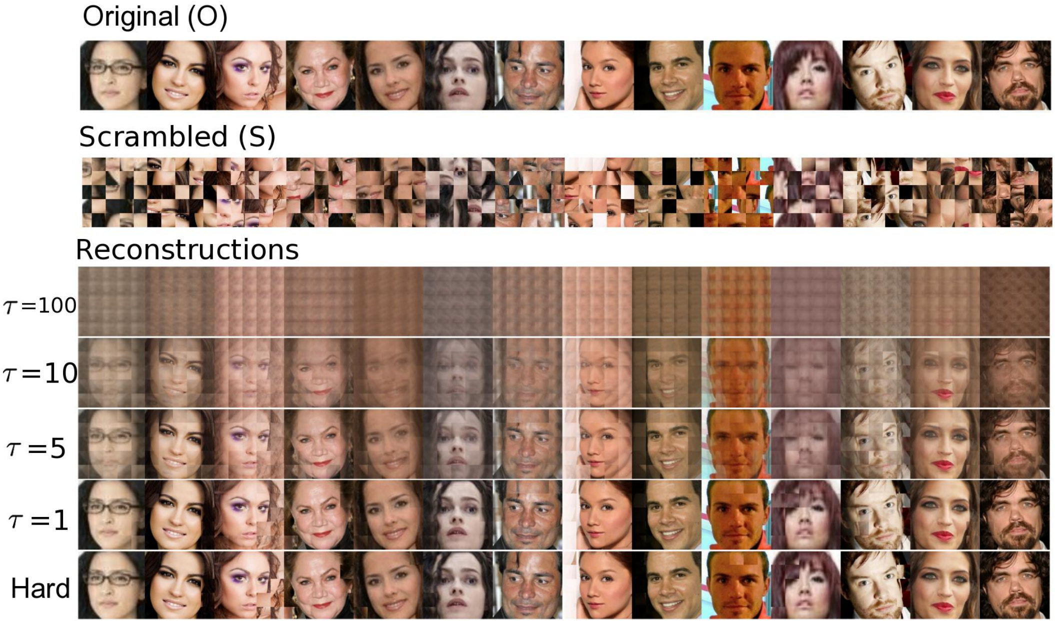
$$M(\Phi) = \lim_{\tau\to 0^+} S(\Phi/\tau)$$

Sinkhorn Networks



$$X_i = P^{-1}_{\theta,\tilde{X}_i} \tilde{X}_i + \varepsilon_i, \quad f(\theta, X, \tilde{X}) = \sum_{i=1}^{M} \left\| X_i - P^{-1}_{\theta,\tilde{X}_i} \tilde{X}_i \right\|^2 \to \min_\theta$$

Experimental Results



Centroid Networks for Few-Shot Clustering and Unsupervised Few-Shot Classification (2019)

Sinkhorn K-means

K-Means. Note that compared to the usual convention, we have normalized assignments $p_{i,j}$ so that they sum up to 1.

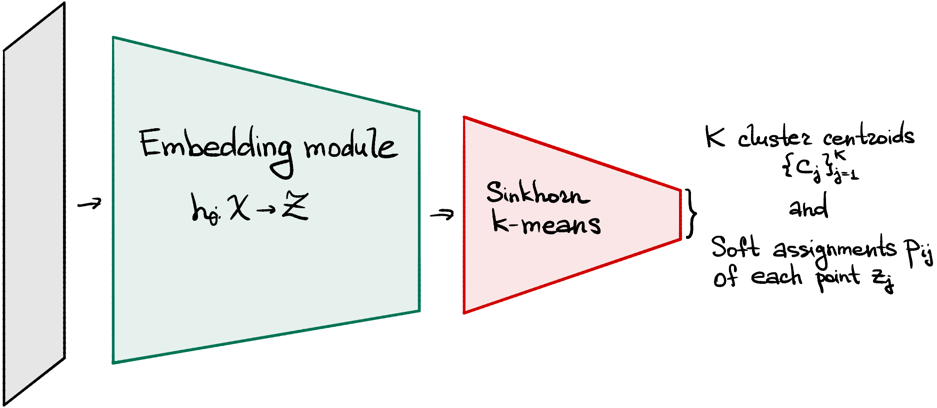
$$\begin{aligned} \text{minimize} \quad & \min_{p,c} \sum_{i=1}^{N} \sum_{j=1}^{K} p_{i,j} ||x_i - c_j||^2 \\ \text{subject to} \quad & \sum_{j=1}^{K} p_{i,j} = \frac{1}{N}, && i \in 1:N \\ & p_{i,j} \in \{0, \tfrac{1}{N}\}, && i \in 1:N, \; j \in 1:K \end{aligned}$$

Sinkhorn K-Means.

$$\begin{aligned} \text{minimize} \quad & \min_{p,c} \sum_{i} \sum_{j} p_{i,j} ||x_i - c_j||^2 - \gamma \underbrace{H(p)}_{\text{entropy}} \\ \text{subject to} \quad & \sum_{j=1}^{K} p_{i,j} = \frac{1}{N}, && i \in 1:N \\ & \sum_{i=1}^{N} p_{i,j} = \frac{1}{K}, && j \in 1:K \\ & p_{i,j} \geq 0 && i \in 1:N, \; j \in 1:K \end{aligned}$$

where $H(p) = -\sum_{i,j} p_{i,j} \log p_{i,j}$ is the entropy of the assignments, and $\gamma \geq 0$ is a parameter tuning the entropy penalty term.

Centroid Networks



Sinkhorn Softmax

- **Softmax conditional:**

$$p_{\boldsymbol{\theta}}(u_i^s = j \mid \boldsymbol{x}_i^s) = \frac{\exp\left\{-\|\boldsymbol{h}_\theta(\boldsymbol{x}_i^s) - \boldsymbol{c}_j\|_2^2 / T\right\}}{\sum_{k=1}^K \exp\left\{-\|\boldsymbol{h}_\theta(x_i^s) - \boldsymbol{c}_k\|_2^2 / T\right\}}$$

- **Sinkhorn conditional:**

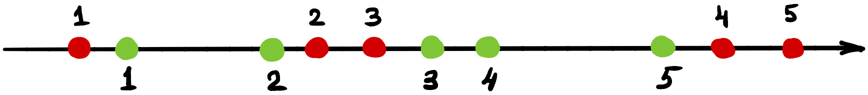
$$p_{\boldsymbol{\theta}}\left(u_i^s = j \mid \boldsymbol{x}_i^s\right) = \frac{p_{i,j}}{\sum_{k=1}^K p_{i,j}}$$

Sliced Wasserstein Distance (2017, CVPR 2018, ICLR 2019)

Wasserstein Distance in 1D

The complexity of computing the Wasserstein distance:

- $d > 1$: $\mathcal{O}\left(n^3 \log n\right)$
- $d = 1$: $\mathcal{O}\left(n \log n\right)$



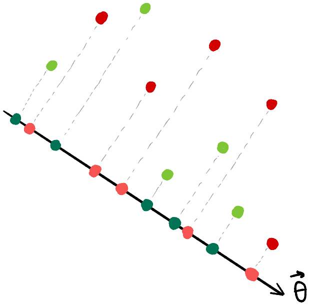
$$W_2^2(X, Y) = \frac{1}{n_{\text{points}}} \|\text{sort}(X) - \text{sort}(Y)\|_1$$

Sliced Wasserstein Distance

Sliced Wasserstein Distance can be defined in the following ways:

- $SW_2^2(X, Y) = \left(\int_{\theta \in \Omega} W_2^2(\theta^\top X, \theta^\top Y)\ \mathrm{d}\theta\right)$,
  where Ω is a unit sphere in $\mathbb{R}^d$
- $SW_2^2(X, Y) = \mathbb{E}_\theta \frac{\|\mathrm{sort}(\theta^\top X) - \mathrm{sort}(\theta^\top Y)\|}{\|\theta\|_1}$,
  where the expectation is taken over the normal distribution in $\mathbb{R}^d$

This distance is usually computed using simple Monte-Carlo methods



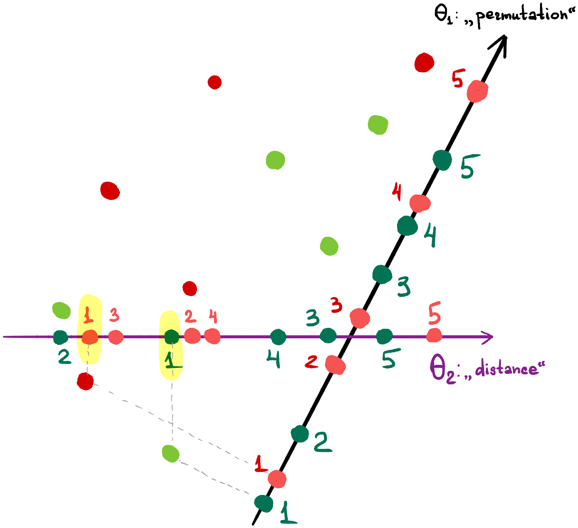
“Orthogonal Estimation of Wasserstein Distances” by Rawland et al. (AISTATS 2019)

Monte-Carlo is biased

We use the same projection to compute ordering and to estimate the distance:

$$SW_2^2(X, Y) \approx \frac{1}{n_{\text{proj}} \cdot n_{\text{samples}} \cdot \|\theta\|} \sum_{i=1}^{n_{\text{proj}}} \|\text{sort}(\theta_i^\top X) - \text{sort}(\theta_i^\top Y)\|_1$$

Projected Wasserstein Distance



Definition 4.1. *Let* $N \leq d$. *The probability distribution* $\mathrm{UnifOrt}(S^{d-1}; N) \in \mathscr{P}((S^{d-1})^N)$ *is defined as the joint distribution of* N *rows of a random orthogonal matrix drawn from Haar measure on the orthogonal group* $\mathscr{O}(d)$. *If* N *is a multiple of* d, *we define the distribution* $\mathrm{UnifOrt}(S^{d-1}; N)$ *to be that given by concatenating* N/d *independent copies of random variables drawn from* $\mathrm{UnifOrt}(S^{d-1}; d)$.

Orthogonal Wasserstein Algorithm: Convergence

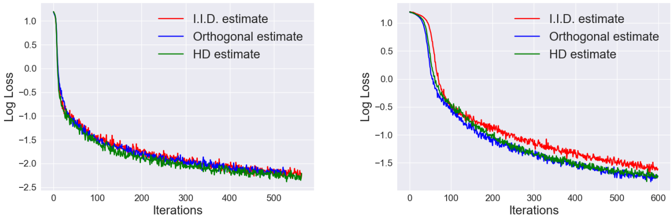


Figure 3: Training curves of Sliced Wasserstein Auto-encoders with three methods to compute Sliced Wasserstein distance: i.i.d. Monte Carlo estimate (red), orthogonal estimate (blue) and HD matrix for orthogonal estimate (green). Vertical axis is the log training loss, horizontal axis is the number of iterations. Left uses a learning rate of $\alpha = 1.0 \cdot 10^{-4}$ and right uses a learning rate of $\alpha = 1.0 \cdot 10^{-5}$.

Questions?