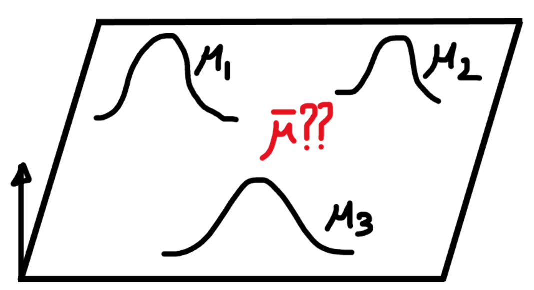
Computing Wasserstein Barycenters: easy or hard?

Jason Altschuler and Enric Boix

Cornell ORIE Young Researchers Workshop 2021

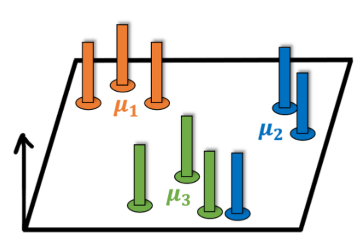
Averaging probability distributions

Fundamental algorithmic primitive in data science and machine learning

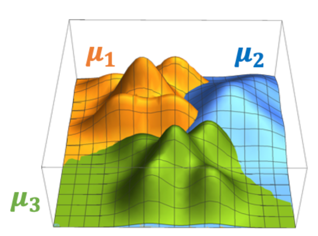


Why average? De-noise, compute exemplar, interpolate, cluster,…

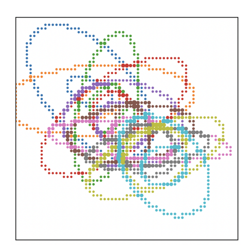
Why distributions? Point clouds in machine learning, posterior distributions in statistics, images in computer vision, object meshes in computer graphics, fMRI scans in neuroscience, …



Ex: point clouds



Ex: statistical models



Ex: images

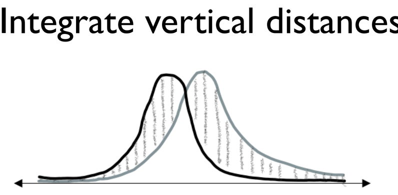
Wasserstein Barycenters

Barycenter: canonical notion of average, given distance.

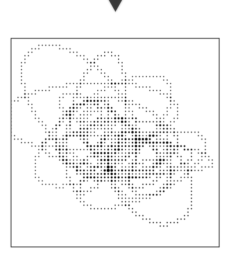
$$\text{argmin}_\nu \sum_{i=1}^{k} d^2(\mu_i, \nu)$$

Using the Wasserstein distance captures the geometry.

Integrate vertical distances

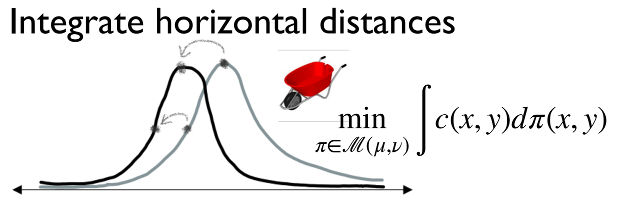


L_p norms, Kullback-Leibler divergence, etc.

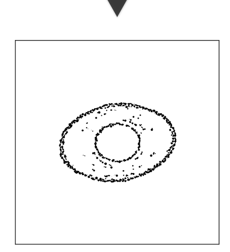


L_2 Barycenter

Integrate horizontal distances



Optimal Transport (aka Wasserstein distance)



Wasserstein Barycenter

(computed with our algorithm)

Algorithmic task

Given: k discrete distributions, each on n points in $\mathbb{R}^d$.

Goal: compute Wasserstein Barycenter in *poly(n,k,d)* time.

Optimization over probability distribution is joint optimization:

1. Mass: easy! Because LP with 1 variable per atom.

2. Support: key issue! Because non-convex & infinite-dimensional.

- Exists barycenter with $O(nk)$ small support.… but how to find?

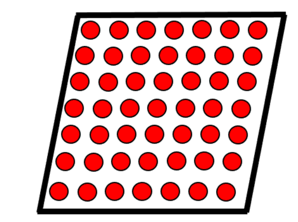
Fundamental question: are Wasserstein Barycenters computable in polynomial time?

Prior work

Despite considerable attention, all prior algorithms have exponential runtime or are heuristic.

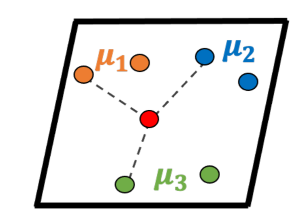
Exponential runtimes in d: restrict to ε-net ➡ runtime factors of $\Omega(1/\varepsilon^d)$

- Intractable beyond low dimension
- Intractable beyond a few digits of accuracy



Exponential runtimes in k: restrict to special n^k points ➡ runtime factors of $\Omega(n^k)$

- Intractable beyond tiny inputs (e.g., k=10 images)



Our contributions

Fixed dimension

Previously: only solve to a few digits of accuracy

Theorem [AB'21a]: For any fixed dimension d, can solve exactly in $poly(n,k)$ time.

- **Enables computing high-precision solutions at previously intractable scales**

High dimension

Previously: only solve for tiny input sizes

Theorem [AB'21b]: Unless P=NP, there is no algorithm with *poly(n,k,d)* runtime.

- **Uncovers robust phenomenon:** hardness extends to approximation, seemingly simple cases, and other optimal transport metrics

Resolves computational complexity of Wasserstein Barycenters

Uncovers "curse of dimensionality" not present for Wasserstein distance computation

Solution strategy

Classical

1. LP reformulation

Steps for solution

2. Implicit LP ideas

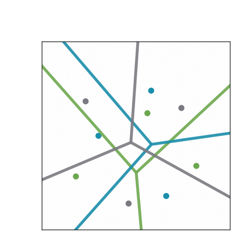
3. Computational geometry & computational complexity ideas

Wasserstein barycenter ⟷ **MOT with barycenter cost** ⟷ **Separation oracle for dual MOT LP** ⟷ **Poly time in fixed dimension** [Intersect power diagrams] / **NP-hard in high dimension** [Reduce from clique]

MOT with barycenter cost — **Pro**: finite size LP; **Con**: n^k variables

Separation oracle for dual MOT LP — **Pro**: combinatorial opt; **Con**: n^k possibilities

Key algorithmic insight: MOT is not a generic LP. Can solve separation oracle efficiently by exploiting the structure of low-dimensional power diagrams.



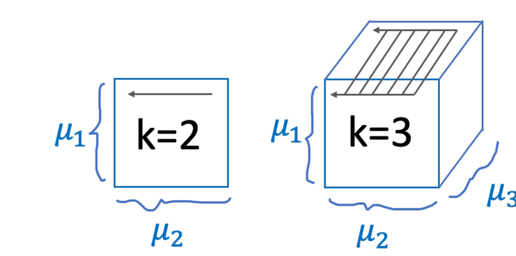
References

[AB'21a] **A** & Boix. Wasserstein barycenters can be computed in polynomial time in fixed dimension. *Journal of Machine Learning Research, 2021.*
[AB'21b] **A** & Boix. Wasserstein barycenters are NP-hard to compute. *SIAM Journal on Mathematics of Data Science, 2021.*
[AB'21c] **A** & Boix. Hardness results for Multimarginal Optimal Transport problems. *Discrete Optimization, 2021.*
[AB'20] **A** & Boix. Polynomial-time algorithms for Multimarginal Optimal Transport problems with structure, *arXiv:2008.03006.*
[ACGS'21] **A**, Chewi, Gerber, Stromme. Averaging on the Bures-Wasserstein manifold: dimension-free convergence of gradient descent. *NeurIPS spotlight, 2021.*

Exponential-size LP reformulation

- **Multimarginal Transportation Polytope:** tensors with fixed marginals

$$\mathcal{M}(\mu_1, \ldots, \mu_k) = \{P \in (\mathbb{R}^n_{\geq 0})^{\otimes k} \ : \ m_i(P) = \mu_i\}$$



- **Multimarginal Optimal Transport:** LP over this polytope

$$\min_{P \in \mathcal{M}(\mu_1,\ldots,\mu_k)} \sum_{x_1,\ldots,x_k} P_{x_1,\ldots,x_k} C_{x_1,\ldots,x_k}$$

- **Fact:** Wasserstein Barycenter optimization is equivalent to MOT with cost

$$C_{x_1,\ldots,x_k} = \min_{y \in \mathbb{R}^d} \sum_{i=1}^{k} \|x_i - y\|^2$$

- **Key issue: this LP has n^k variables**
  - n^k is humongous (e.g., $k = 100$ images)
  - Can't even store cost tensor C or solution P. And even if you could, can't solve…

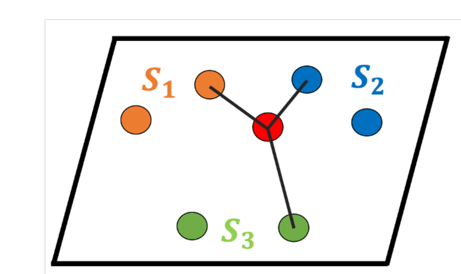
Algorithm for fixed dimension (simplified)

Dual separation oracle (simplified)

Given: k sets of n points in $\mathbb{R}^d$

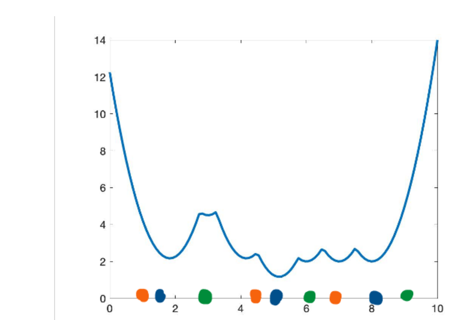
Compute: $\min_{y \in \mathbb{R}^d} \min_{x_1 \in S_1, \ldots, x_k \in S_k} \sum_{i=1}^{k} \|x_i - y\|^2$

Trivial given y: take closest point to y.



But, how to optimize non-convex $F(y)$?

- Piecewise convex on finitely many "pieces"
- Naive bound is n^k pieces (1 per tuple $x_1, \ldots, x_k$)



Key lemma: For fixed d, only *poly(n,k)* pieces!

Key proof technique: Hyperplanes partition $\mathbb{R}^d$ into few regions.

Algorithm: Enumerate pieces. Easily optimize y on each piece. Return best.

My related line of work

- **Q:** We showed NP-hard in high dim… **What about simple restricted settings?**
- **A: Yes**, the standard algorithm used for Gaussians runs in polynomial time [ACGS'21]
  - **Key obstacle:** analyze gradient descent on non-convex function
  - **Key insight:** geodesic convexity of $-\sqrt{\lambda_{\min}}$ and $\sqrt{\lambda_{\max}}$ in Bures-Wasserstein space

- **Q:** This is all for Wasserstein Barycenters… **General algorithmic understanding?**
- **A: Yes**, polynomial-time algorithms for wide range of Multimarginal Optimal Transport problems with "structure" [AB'20]. Also understand fundamental limitations [AB'21c].
  - **Key obstacle:** different separation oracles require very different algorithms
  - **Key insight:** unified algorithmic framework that captures nearly all applications