Wasserstein Wormhole: Scalable Optimal Transport Distance with Transformer

Doron Haviv [1 2] **Russell Zhang Kunes** [1 3] **Thomas Dougherty** [1 2] **Cassandra Burdziak** [1] **Tal Nawy** [1]
Anna Gilbert [4] **Dana Pe'er** [1 5]

Abstract

Optimal transport (OT) and the related Wasserstein metric (W) are powerful and ubiquitous tools for comparing distributions. However, computing pairwise Wasserstein distances rapidly becomes intractable as cohort size grows. An attractive alternative would be to find an embedding space in which pairwise Euclidean distances map to OT distances, akin to standard multidimensional scaling (MDS). We present Wasserstein Wormhole, a transformer-based autoencoder that embeds empirical distributions into a latent space wherein Euclidean distances approximate OT distances. Extending MDS theory, we show that our objective function implies a bound on the error incurred when embedding non-Euclidean distances. Empirically, distances between Wormhole embeddings closely match Wasserstein distances, enabling linear time computation of OT distances. Along with an encoder that maps distributions to embeddings, Wasserstein Wormhole includes a decoder that maps embeddings back to distributions, allowing for operations in the embedding space to generalize to OT spaces, such as Wasserstein barycenter estimation and OT interpolation. By lending scalability and interpretability to OT approaches, Wasserstein Wormhole unlocks new avenues for data analysis in the fields of computational geometry and single-cell biology. Software is available at http://wassersteinwormhole.readthedocs.io/en/latest/.

[1]Computational and Systems Biology Program, Sloan Kettering Institute, Memorial Sloan Kettering Cancer Center [2]Tri-Institutional Training Program in Computational Biology and Medicine, Weill Cornell Medicine [3]Department of Statistics, Columbia University [4]Department of Mathematics and Statistics, Yale University [5]Howard Hughes Medical Institute. Correspondence to: Dana Peer <peerd@mskcc.com>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

1. Introduction

Many problems in statistical learning boil down to computing and comparing probability distributions that fit empirical data, and different approaches vary in their attention to distribution shape, spread, and degree of parameterization. Optimal transport (OT) methods attempt to find an ideal pairing between individual data points from two distributions (Villani et al., 2009) by representing distances as the minimum cost of mapping between (weighted) point clouds. While original OT solvers, which rely on linear programming, scale poorly (Peyré et al., 2019), newer approaches can reduce the computational complexity and generalize OT to new domains such as computational biology (Nitzan et al., 2019; Schiebinger et al., 2019; Bunne et al., 2023), numerical geometry (Su et al., 2015), chemistry (Wu et al., 2023), and image processing (Feydy et al., 2017).

We are specifically interested in massive cohorts of empirical distributions, each with up to thousands of points. The computational complexity of finding the Wasserstein distance between two point clouds with Sinkhorn's algorithm (Cuturi, 2013), the current favored (Flamary et al., 2021) OT solver, is $O(n^2 \cdot I)$, where n is the size of each point cloud and I is the number of Sinkhorn iterations until convergence. As we must repeat Sinkhorn's algorithm for all pairs in a cohort, requiring $O(N^2 \cdot n^2 \cdot I)$ time to analyze N distributions, current OT algorithms are unable to analyze large cohorts despite improvements in scalability.

Here, we present Wasserstein Wormhole, an algorithm that represents each point cloud as a single embedded point, such that the Euclidean distance in the embedding space matches the OT distance between point clouds. In Wormhole space, we compute Euclidean distance in $O(d)$ time for an embedding space with dimension d, which acts as an approximate OT distance and enables Wasserstein-based analysis without expensive Sinkhorn iterations.

Point clouds contain a variable number of unordered samples, hence neural architectures that require a fixed-sized or ordered input are not suited to encode them. On the contrary, transformers (Vaswani et al., 2017) are uniquely poised to do so, due to their permutation equivariance, indifference to input length, and ability to model relationships between samples in each point cloud via attention.

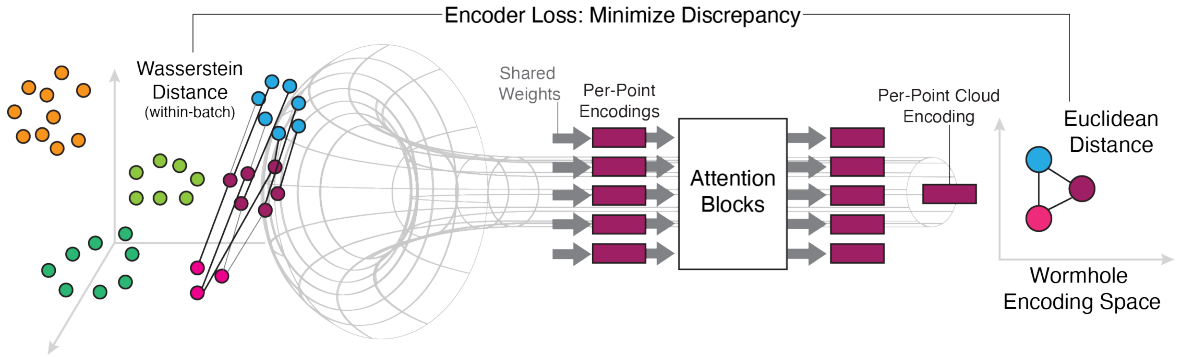

Figure 1. **Schematic of Wasserstein Wormhole**. Empirical distributions (point clouds) are passed through a transformer to produce per point-cloud vector embeddings such that the Euclidean distance between embeddings match the pairwise Wasserstein distance between point clouds. Since computation of OT distances is laborious, Wormhole is optimized by mini-batches to minimize the discrepancy between the embedding pairwise distances and the pairwise Wasserstein distances of the batch point clouds. The Wormhole decoder (not shown) is a second transformer trained to reproduce the input point clouds from the embedding by minimizing the OT distance between input and output.

In addition to an encoder that embeds point clouds, Wormhole trains a decoder that reconstructs point clouds from their embedding. The decoder facilitates interpretability with its ability to explicitly compute Wasserstein barycenters (Cuturi & Doucet, 2014) or interpolate between point clouds (Chewi et al., 2021).

The problem solved by Wormhole is analogous to multidimensional scaling (Torgerson, 1952). Both methods find an embedding that preserves distances between samples. While several distance metrics are provably capable of being represented in this manner, it has been shown that a host of other "non-Euclidean" metrics, including Wasserstein, cannot be exactly equated with Euclidean distance. Counter-intuitively, classical MDS on non-Euclidean distance matrices becomes more inaccurate as embedding dimensions increase (Sonthalia et al., 2021).

Inspired by (Sonthalia et al., 2021), we extend MDS theory to derive upper and lower bounds for the error incurred by embedding non-Euclidean distances, such as Wasserstein. We further describe a projected gradient descent (PGD) (Boyd & Vandenberghe, 2004) algorithm for MDS which is guaranteed to converge to the global optimum for any distance matrix. On small examples where it is feasible to compute bounds and apply PGD, we find that Wormhole approaches the optimal solution and falls within our derived bounds.

2. Prior Work

There have been many prior works on learning from point cloud data (Guo et al., 2020), often involving OT-based solutions. PointNet (Garcia-Garcia et al., 2016; Qi et al., 2017) introduced the idea of permutation-invariant, fully-connected neural networks to classify and segment 3D point-clouds, which was later extended into transformer-based networks (Lee et al., 2019). In the context of OT, several works have noted the expressiveness of Wasserstein space compared to euclidean space (Frogner et al., 2019; Kataeri et al., 2024; Liu et al., 2022), and have developed methods to embed point-clouds into Wasserstein space, from which relevant information can be more reliably extracted. These works are in essence the inverse of Wormhole, as their target space is Wasserstein, in contrast to Wormhole's Euclidean target space.

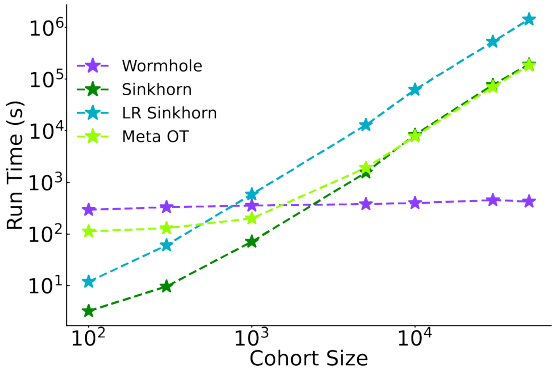

Figure 2. **Benchmarking run-time of Wormhole against other OT acceleration algorithms**. Sampling cohorts of different sizes from the MNIST dataset, we measured the time required for current acceleration algorithms (with GPU implementation in JAX-OTT) and Wormhole to compute or approximate the pairwise Wasserstein matrix. Other methods are more appropriate at tiny cohorts, as they do not require training a parametric model. However, even in cohorts with relatively few samples, Wormhole is superior and no other method can scale to complete datasets, requiring weeks of compute time, even on a fully-utilized 80GB GPU.

Wasserstein Wormhole belongs to a class of algorithms that exploit structures within distributions or relationships between cohorts of them to accelerate or approximate OT. Low-Rank (LR) Sinkhorn (Scetbon et al., 2021) factorizes

	Dataset Parameters			OT Correlation				Label Accuracy			
Name	Cohort Size	Cloud Size	Dimension	Wormhole	DiffusionEMD	DWE	Fréchet	Wormhole	DiffusionEMD	DWE	Fréchet
MNIST	70,000	105	2	**0.98**	0.845	0.92	0.86	**0.98**	0.84	0.97	0.49
Fashion-MNIST	70,000	356	2	**0.99**	0.97	0.98	0.99	**0.87**	0.77	**0.87**	0.73
ModelNet40	12,311	2048	3	**0.99**	0.85	0.97	0.95	**0.86**	0.69	0.78	0.61
ShapeNet	15,011	2048	3	**0.99**	0.81	0.97	0.97	**0.98**	0.94	0.97	0.92
Rotated ShapeNet (GW)	15,011	512	3	**0.98**	NA	NA	0.68	**0.82**	NA	NA	0.16
MERFISH Cell Niches	256,058	11	254	**0.97**	OOM	OOM	OOM	**0.98**	OOM	OOM	OOM
SeqFISH Cell Niches	57,407	29	351	**0.98**	OOM	OOM	OOM	**0.97**	OOM	OOM	OOM
scRNA-seq Atlas	2,185	69	2500	**0.98**	OOM	OOM	OOM	**0.96**	OOM	OOM	OOM

Table 1. Summary of each dataset used in this study and results from every benchmarked algorithm. Cloud size denotes the median number of points in each point cloud for every dataset. Pearson correlations are computed between true and estimated Wasserstein distances according to pairwise matrices of 10 instances of 128 randomly chosen point clouds from each dataset (OT Correlation). The mean squared error (MSE) follows the same trend with Wormhole outperforming competing methods (Table 3). Object classification accuracy is based on the embeddings of each algorithm. After computing encodings, a classifier was trained on the training set embeddings and we denote the classifier's accuracy on the test set (Label Accuracy). Rotated ShapeNet is a version of the standard ShapeNet dataset where each sample is randomly rotated in 3D, subsampled by a factor of 4, and embedded via Gromov-Wasserstein (GW). We divide the results into Wasserstein on 2D and 3D point-clouds, Gromov-Wasserstein on rotated data with arbitrary orientation and high-dimensional datasets. Full implementation details are described in the supplement (Section 11). NA, Not Applicable. OOM, Out-Of-Memory.

the pairing matrix into a low-rank composition, assuming the transport plan between two distribution is structured, to enable faster computation of Sinkhorn iterations. Convolutional and Geodesic Sinkhorn (Bachmann et al., 2022; Solomon et al., 2015) construct solvers which leverage the underlying geometry of the point-cloud space to simplify OT optimization. MetaOT (Amos et al., 2022) recognizes that convergence of Sinkhorn depends greatly on quality of initialization and fits a neural network which learns an initial Sinkhorn potential for a given pair of input distributions. Due to its fully-connected architecture, MetaOT is constrained to distributions which share supports, and cannot be applied to arbitrary point-clouds. Although these methods can speed up the computation of an individual OT problem, none alleviate the need for many expensive pairwise computations, and are far slower than Wormhole on large cohorts (Figure 2).

Previous works have also employed Wasserstein preserving euclidean embeddings in the pursuit of scaling OT to large cohorts (Tong et al., 2021), some using neural network models (Courty et al., 2017; Sehanobish et al., 2020). DWE (Courty et al., 2017) trains a Siamese neural network to match distances, and DiffusionEMD (Tong et al., 2021) extends the wavelet-based approximation of EMD (Shirdhonkar & Jacobs, 2008) onto graph domains. Unlike our Wormhole approach, which applies to point clouds in any dimension, DWE is constrained to small 2D images or 3D grids, while scalability of DiffusionEMD is limited to distributions that share supports, such as grayscale images or voxelized spaces. We benchmark Wormhole against DiffusionEMD & DWE, and show that our method is superior at matching OT distances and deriving insights from embeddings on low-dimensional point clouds. On general datasets, Wormhole is the only embedding method which can scale to high-dimensions.

3. Preliminaries

3.1. Optimal Transport

We follow the formulation of optimal transport presented in (Peyré et al., 2019). Given two discrete distributions $P_x = \sum_{i=1}^{n} \mu_i \delta_{x_i}$ and $P_y = \sum_{i=1}^{m} \nu_i \delta_{y_i}$ (δ is the Dirac function), their entropically regularized OT distance is:

$$W_\varepsilon(P_x, P_y) = \min_{P \in U(\nu,\mu)} \sum_{i,j} c(x_i, y_j) P_{i,j} \; - \varepsilon H(P) \quad (1)$$

Where $U(\nu, \mu) := \{P \in R_+^{n,m} : \sum_j P_{i,j} = \nu_i, \sum_i P_{i,j} = \mu_j\}$ is the set of all valid pairing matrices, $c(x_i, y_j)$ is the ground cost function between points in P_x and P_y, usually chosen to be the Euclidean distance, and $H(P) = -\sum_{i,j} P_{i,j}(\log P_{i,j} - 1)$ is the entropic term. While the unregularized problem incurs a heavy computational burden for distributions with large supports or in high dimensions, W_ε can be computed via a series of GPU-accelerable vector-matrix products known as Sinkhorn iterations (Cuturi, 2013).

Due to the regularization term, the W_ε distance between a distribution and itself is not 0, which can introduce biases when used to optimize generative models. To allow for effective optimization while retaining the computational advantages of entropic regularization, a common alternative is the Sinkhorn Divergence (Genevay et al., 2018), which subtracts the self-transport costs from W_ε:

$$S_\varepsilon(P_x, P_y) = W_\varepsilon(P_x, P_y) - \frac{W_\varepsilon(P_x, P_x) + W_\varepsilon(P_y, P_y)}{2} \quad (2)$$

3.2. Multi-dimensional Scaling

Given the pairwise Wasserstein distance matrix between a cohort of distributions $D_{i,j} = W_\varepsilon(X_i, X_j)$, we seek an embedding per distribution, such that pairwise Euclidean distance between embeddings match the OT distance $\|\alpha_i - \alpha_j\|_2^2 \approx W_\varepsilon(X_i, X_j))$. In essence, this describes the MDS problem with the critical difference that for large cohorts, it is computationally infeasible to compute the OT distance between all pairs of distributions. Still, MDS can provide key theoretical insights for our setting.

Definition 3.1 (Euclidean distance matrix (EDM))**.** An $N \times N$ distance matrix D is Euclidean if there exists a set of points $X = \{x_1, x_2, \ldots, x_N\}$ in some dimension $x_i \in R^d$ such that:

$$\forall i, j: \; D_{i,j} = \|x_i - x_j\|_2^2 \tag{3}$$

From (Gower, 1985), a matrix D is a Euclidean Distance Matrix if and only if all of its values are non-negative, it is hollow (diagonal elements are 0) and the criterion matrix F is positive-semidefinite (PSD):

$$F = -JDJ \tag{4}$$

Where $J = I_N - \frac{\mathbf{1}_N \mathbf{1}_N^\top}{N}$ is the centering matrix and 1_N is a vector of length N filled with ones.

Wasserstein distance is not Euclidean (see section 8.3 in (Peyré et al., 2019) and (Naor & Schechtman, 2007)), and there are many examples of cohorts of distributions (Figures S1,S2) wherein pairwise OT distance matrices do not fit the criteria in equation 4. Unlike the Euclidean scenario where classical MDS (cMDS) improves as embedding size grows, on non-Euclidean distance matrices the error behaves counter-intuitively and worsens as dimension increases (Sonthalia et al., 2021). This motivates us to seek a description of the theory underpinning the embeddings of non-Euclidean distances.

4. Wasserstein Wormhole

The complexity of modern OT solvers is quadratic with respect to the number of points in a point cloud; thus, it is only straightforward to compute pairwise Wasserstein distance for small cohorts of empirical distributions. Indeed, naive approaches will not even scale to standard datasets, such as the 12,311-point-cloud ModelNet40 benchmark (Wu et al., 2015).

This problem calls for a stochastic approach. Instead of directly assigning each point cloud with an embedding, we propose learning a parametric encoding function. Formally, the dataset in our setting is a cohort of point clouds $\Omega = \{X_1, X_2, X_3, \ldots X_N\}$, where each point cloud X_i contains n_i points in d dimensions. Our goal is to find a mapping function $T(X)$ that encodes point clouds into embeddings α, such that Euclidean distances between encodings approximate the OT distances been point clouds. By computing the pairwise Wasserstein distance matrix for small mini-batches, we can train $T(X)$ so that encoding distances directly match Wasserstein.

Since $T(X)$ is applied to point clouds, it should contain a few key attributes:

- Wasserstein distance is invariant with respect to the order of samples within point clouds ($W(X, PY) = W(X, Y)$, P is a permutation matrix). The encoding must therefore also be permutation invariant: $T(X) = T(PX)$.
- Point clouds contain varying numbers of samples. $T(X)$ must be able to generalize to arbitrary cloud sizes while modeling relationships between the points within them.

Transformers—specifically, stacked layers of multi-head attention—fit both of these criteria. Although they are commonly associated with sequential data such as words in a sentence, order information is provided as input to the model. Without the positional embeddings, attention mechanisms (and therefore transformers) are permutation equivariant (proof in supplementary section 8). By performing an invariant function such as averaging on the final set of embeddings, this produces a mapping $T(X)$ that is also invariant by design.

Algorithm 1 Wasserstein Wormhole At each training step, a mini-batch of point clouds is randomly sampled from the cohort. Wormhole is optimized so that pairwise distance between mini-batch embeddings match their OT distance and output decodings reconstruct input point clouds.

Input: point clouds $\Omega = \{X_i\}_{i=1}^N$, initialized encoder T and decoder G networks, batch size B, learning rate ε_t
repeat
  Sample B point clouds $\{x_i\}_{i=1}^B$ from Ω
  Calculate encodings $\alpha_i = T(x_i)$
  Compute pairwise OT in batch $D_{i,j} = S_\varepsilon(x_i, x_j)$
  Evaluate stress $\mathcal{L}_{enc} = \sum_{i,j} (\|\alpha_i - \alpha_j\|_2^2 - D_{i,j})^2$
  Produce decodings $\hat{x}_i = G(\alpha_i)$
  Evaluate decoding error $\mathcal{L}_{dec} = \sum_i S_\varepsilon(x_i, \hat{x}_i)$
  Update encoder $T \leftarrow T - \varepsilon_t \nabla(\mathcal{L}_{enc} + \mathcal{L}_{dec})$
  Update decoder $G \leftarrow G - \varepsilon_t \nabla(\mathcal{L}_{dec})$
until Convergence

By training the encoder network $T(X)$ (Figure 1, Algorithm 1), we can produce encodings which allow for OT-based analysis of point clouds in linear time, enabling methods such as clustering and visualization (Bachmann et al., 2022)

on the entire cohort. However, the encoder alone does not provide a clear interpretation of these analyses.

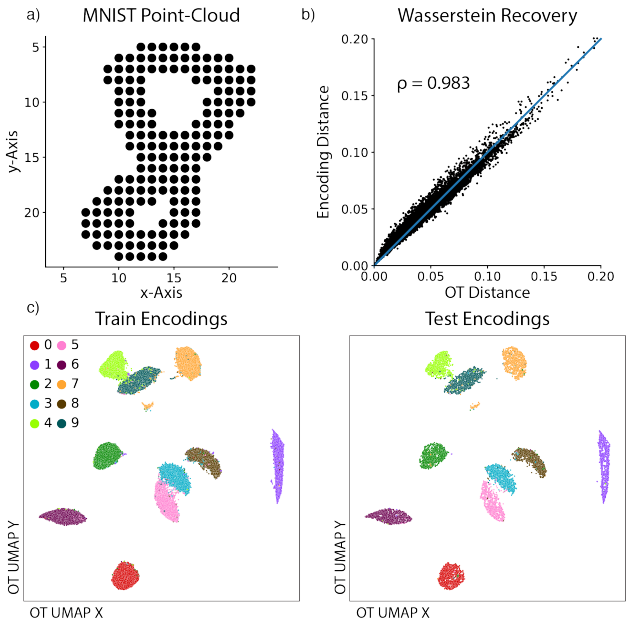

Figure 3. **Wormhole on MNIST point clouds. a**. Example of a point cloud produced by thresholding MNIST images. **b.** Retrieval of the Wasserstein distance in Wormhole embedding space. From a random sample of 128 point clouds from the test set, we denote the correlation between the true pairwise OT distance and the Euclidean distance between their embeddings. The line $y = x$ is drawn in blue for reference. **c.** 2D UMAP visualization of Wormhole embeddings of the training and test set point clouds, which recapitulates ground-truth image digit without utilizing label information during training.

To imbue the embeddings with interpretability, we include a decoder network $G(\alpha)$ which takes encodings as input and is simultaneously trained to reproduce the input point clouds by minimizing the Wasserstein distance loss. Unlike standard autoencoders, which only train the full network to match outputs to inputs, Wormhole imposes a loss on both the encodings and decodings.

5. Embedding Non-Euclidean Distance Matrices

Given a distance matrix D sized $N \times N$, we are interested in how well it can be approximated with a series of embeddings $\{\alpha_i\}_{i=1}^N$ in d dimensions $\alpha_i \in R^d$. This corresponds to the optimization problem of minimizing the cumulative *stress*:

$$\mathcal{L} = \min_{\{\alpha_1, \alpha_2, \ldots, \alpha_N\}} \sum_{i,j} (\|\alpha_i - \alpha_j\|_2^2 - D_{i,j})^2 \tag{5}$$

In the case where D is Euclidean, by definition there exists a set of embeddings in dimension $d = N$ where the distance matrix is perfectly reconstructed and $\mathcal{L} = 0$. Wormhole, however, is trained to minimize the *stress* relative to the Wasserstein distance matrix, which is generally *not* Euclidean. It is currently unclear to what extent D can be represented with embeddings when it is not Euclidean (Sonthalia et al., 2021).

As formulated in equation 5, this problem is non-convex and therefore difficult to study. However, when we instead reparameterize the objective to be a function over the distance matrix produced from the embedding $\hat{D}_{i,j} = \|\alpha_i - \alpha_j\|_2^2$ and add relevant constraints for $\hat{D}$ to be Euclidean, the problem is rendered convex :

$$\mathcal{L} = \min_{\hat{D}} \sum_{i,j} (\hat{D}_{i,j} - D_{i,j})^2$$

$$\text{s.t. } C_1\text{: } -J\hat{D}J \text{ is PSD, } C_2\text{: } \hat{D} \text{ is hollow and } C_3\text{: } \hat{D}_{i,j} \geq 0. \tag{6}$$

Since all constraints are convex sets, we could construct a projected gradient descent approach (Algorithm 2) that is guaranteed to converge to the global minimum $\boldsymbol{\mathcal{L}}^*$. Furthermore, we derive novel theoretical lower and upper bounds for the global minimum $\boldsymbol{\mathcal{L}}^*$.

Theorem 5.1 (Lower Bound)**.** *For a given distance matrix D and the eigendecomposition $\{\lambda_i, v_i\}_{i=1}^N$ of its criterion matrix $F = -JDJ$, the optimal stress is greater or equal to:*

$$\boldsymbol{\mathcal{L}}^* \geq \mathbf{L} = \sum_{i:\lambda_i<0} \lambda_i^2 \tag{7}$$

This bound also appears in (Sonthalia et al., 2021), which analyzed pathologies of cMDS on non-Euclidean distance matrices.

Theorem 5.2 (Upper Bound)**.** *The optimal stress is less than or equal to:*

$$\boldsymbol{\mathcal{L}}^* \leq \mathbf{U} = \sum_{i,j} (\Delta g_i + \Delta g_j)^2 + \sum_{i:\lambda_i<0} \lambda_i^2 \tag{8}$$

Where:

$$\Delta g_i = \frac{1}{2} \sum_{j:\lambda_j<0} \lambda_j \cdot v_{i,j}^2 \tag{9}$$

And $v_{i,j}$ is the j-th element of the i-th eigenvector of F.

The prominence of the negative spectrum of the criterion matrix can be thought of as a quantitative measure of how close D is to being Euclidean. As we expect, the more minute the negative eigenvalues of F (as in Figure S2), the smaller both the lower and upper bounds become, and we can anticipate that the embeddings will increase in quality.

The derivation of the lower bound comes from removing the constraints C_1 and C_2 on hollowness and non-negativity of $\hat{D}$ and computing the projection operator onto C_1 which is the space of matrices where $-JDJ$ is positive semi-definite (subsection 9.1). Although it is similar to the standard projection onto PSD and incurs the same truncation error, it has additional terms due to the double-centering transform.

Algorithm 2 Theoretically Optimal Embedding of non-Euclidean Distance Matrices with Projected Gradient Descent (**PGD**)

Input: Distance Matrix D, Initial Solution D'_0, Learning Rate ε_t
repeat
  $D^*_{t+1} = D'_t - \varepsilon_t \nabla(\|D'_t - D\|^2_F)$ {GD Step}
  $D'_{t+1} = \text{proj}_{C_1 \cup C_2 \cup C_3}(D^*_{t+1})$ {Dykstra}
until Convergence
RETURN $X = \text{cMDS}(D')$ {Embed D' with MDS}

This new projection operator, along with the simple projections onto C_2 (hollow matrices, by setting the diagonal to 0) and C_3 (non-negative matrices, by thresholding), allows us to formulate a PGD (Boyd & Vandenberghe, 2004) scheme (Algorithm 2) to embed non-Euclidean distance matrices. This algorithm projects onto $C_1 \cup C_2 \cup C_3$ using Dykstra iterations (Boyle & Dykstra, 1986) and is guaranteed to find the optimum for equation 6, unlike other MDS approaches such as cMDS or SMACOF (De Leeuw & Mair, 2009). The PGD algorithm requires knowing the full pairwise distance matrix and is therefore infeasible in our setting of massive cohorts. However, for small cohorts, the theoretical assurances of PGD provide a useful benchmark for Wormhole.

Our analysis indicates that in real-world datasets, the Wormhole approximation is highly accurate and commensurate with theoretical values. Although this assessment is based on a small subsample of the full dataset, we assume that if random subsets of the larger dataset can be well embedded, the embedding of the entire dataset is reliable (Table 2). Full derivation of the theoretical bounds appear in Supplementary Sections 9 & 10.

6. Experiments

We demonstrate Wasserstein Wormhole on datasets representing multiple contexts where OT is frequently applied. We applied Wormhole to 2D point clouds derived from the MNIST and Fashion-MNIST datasets, and to 3D CAD models from ModelNet40 and ShapeNet (Chang et al., 2015). As a high-dimensional applications, we used Wormhole to embed multicellular niches (Haviv et al., 2023) from two high-resolution spatial transcriptomics datasets (Zhang et al., 2021; Lohoff et al., 2022), along with groups of cells ('MetaCells' (Persad et al., 2023)) from a scRNA-seq atlas of patient response to COVID (Stephenson et al., 2021).

Dataset	Lower	Upper	PGD	Wormhole	cMDS
Simplex (35)	0.765	6.369	1.117	1.420	7.134
Gaussian (128)	0.129	2.552	0.168	0.401	2.681
MNIST (256)	0.042	0.616	0.058	0.100	0.657

Table 2. Comparison between embedding algorithms and theoretical bounds on small datasets where full Wasserstein distance matrices are realizable. Values are the *stress* between the embedding and true distances and parentheses denote the size of each dataset. Classical MDS produces far from optimal errors, falling outside our bounds. Wormhole training is stochastic, yet it still converges between the error bounds and approaches the error obtained by the theoretically optimal PGD. Both PGD and Wormhole embeddings were well correlated ($\rho = 0.99$) with OT distances.

For each dataset, we quantitatively benchmarked Wormhole by measuring how well the learned embeddings matched the true OT distance and ground-truth labels. Since the full Wasserstein distance matrix is unattainable, for each dataset we randomly sampled 10 instances of $n = 128$ point clouds and compared the $n \times n$ OT distance matrix to the embedding Euclidean distance. All the datasets are labeled, allowing us to measure the classification accuracy based on held-out test-sets for each.

We benchmarked our Wormhole against DiffusionEMD and Deep Wasserstein Embedding (DWE), both sharing Wormhole's goal of finding OT preserving embeddings. We also compare against a constructed baseline we dub *Fréchet* (Dowson & Landau, 1982) in which we use the closed-form W_2 distance for Gaussian distributions. For each point cloud, we calculated the mean (μ) and covariance matrix (Σ) of its samples and use the following formula to approximate the OT distance between each point cloud pair:

$$W_2^2 \approx \|\mu_i - \mu_j\|^2 + \text{Tr}\left(\Sigma_i + \Sigma_j - 2\sqrt{\Sigma_i \Sigma_j}\right) \quad (10)$$

The following sections present vignettes of Wormhole from several dataset, and results are summarized in Table 1. Wormhole outperforms all competing approaches and uniquely scales to massive & high-dimensional datasets such as the the 256,058-sample, 254-dimensional *MERFISH Cell Niches* cohort. We specifically highlight the improvement of Wormhole over DWE, which similarly learns a neural network parametric embedding. The transformer backbone of Wormhole is superior to DWE's CNN in low-dimensional datasets, while also proving to be essential in scaling Wormhole beyond 3D data.

6.1. MNIST (Touchstone)

We applied Wormhole to the MNIST dataset of hand-written digits, which consists of 70,000 samples of 28×28 pixel

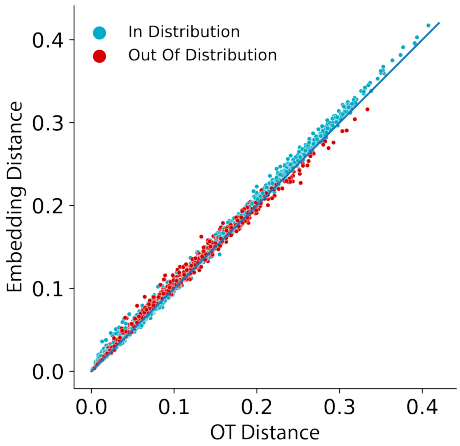

Figure 4. **Generalization to OOD on Fashion-MNIST**. During training, we held out out point clouds labeled 'Bag'. Performance was slightly lower compared to observed classes, especially for larger Wasserstein distances, as the MSE for OOD samples was $4.38 \cdot 10^{-5}$ as opposed to $3.37 \cdot 10^{-5}$ for observed samples. While not trained on any 'Bag' labeled point clouds, their encodings still largely agree with true OT distance, producing a Pearson correlation of $\rho = 0.995$ and label accuracy of 0.97.

grayscale images. We converted each sample to a point cloud by thresholding and selecting the set of pixel coordinates that passed the threshold (Figure 3a). While MNIST is considered small by modern machine-learning standards, straightforward computation of all $\binom{70000}{2} = 2.45 \cdot 10^9$ pairwise Wasserstein distances using current OT solvers is far from computationally feasible and would require a week of JAX-OTT (Cuturi et al., 2022) on an 80GB GPU (Figure 2).

By contrast, training Wormhole with default parameters (Supplementary Section 11) requires a small fraction of OT computations (approximately a 2000-fold reduction) compared to the naive approach, with a total runtime of only 10 min. Despite the speedup, Wormhole still achieves remarkable performance, learning common patterns within the MNIST dataset and significantly outperforming contemporary approaches. Pairwise distances between point cloud encodings match their true OT distances and represent underlying digits (Figure 3b,c).

6.2. Fashion-MNIST (Out-Of-Distribution)

Similar to MNIST, FashionMNIST contains 70,000 images that each measure 28×28 pixels, but the images depict articles of clothing instead of digits. After thresholding FashionMNIST images to produce point clouds, Wormhole was able to accurately recover OT distances and sample labels (Table 1) in the embedded space.

For a model to be deployed confidently, it is imperative that it can generalize to out-of-distribution (OOD) examples. To evaluate the capabilities of Wormhole on OOD data, we hid all examples labeled 'Bag' during training and measured the quality of encodings. We found that Wormhole preserves the OT metric (Figure 4). In addition, when classifying test-set encodings based on in-distribution and OOD samples, accuracy does not diminish on the held-out class. Regardless of whether or not 'Bag' samples were included in Wormhole training, their test-set accuracy was consistently 0.97. We repeat this experiment for all 10 classes in the FashionMNIST cohort and find that results are robust, having an overall OOD error of $3.14 \cdot 10^-5 \pm 2.22 \cdot 10^{-5}$ and correlation of 0.998 ± 0.001.

6.3. ModelNet40 (Variational Wasserstein)

The ModelNet40 dataset comprises point clouds from 3D synthetic CAD models of 40 diverse object classes. While each point cloud is more complex than the image datasets, containing thousands of points as opposed to hundreds, Wormhole was able to produce accurate embeddings in 30 GPU min of training time (Table 1).

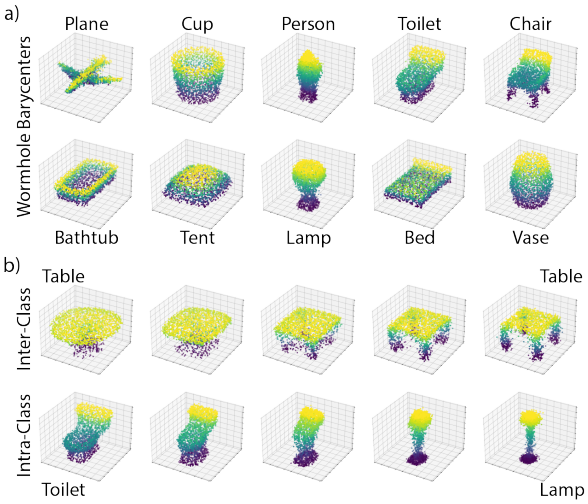

Figure 5. **Wormhole Variational Wasserstein. a.** Mean encodings of each class were passed through the decoder to visualize their Wasserstein barycenters. Working in Wormhole space provides a quick and straightforward proxy for OT analysis, and recovered decodings reproduce the underlying class well. For brevity, we show 10 out of the 40 classes in from ModelNet40. **b.** Wormhole interpolation between two point clouds within the same class (top) and across different classes (bottom). Intermediate point clouds are decodings of linear interpolations between embeddings.

As a more challenging test case, we assessed the ability of the Wormhole decoder to interpret the embeddings and solve variational Wasserstein problems. The Wasserstein barycenters for a set of distributions is their average OT distance. Calculating barycenters is computationally challenging even in a small cohort. Wormhole embedding space is a Euclidean approximation of Wasserstein and barycen-

ters of a set can be computed by calculating their encoding average and decoding it (Figure 5a).

Using Wormhole, we can also smoothly transition between point clouds via Wasserstein interpolation in the embedded space. Interpolating between distributions in Wasserstein space involves solving a weighted version of the barycenter problem as many times as there are intermediate samples. With Wormhole, we can leverage the Euclidean approximation of its latent space and directly decode linear interpolations of embeddings (Figure 5b). Despite interpolated embeddings lying outside of Wormhole training distribution, their decodings reliably visualize point cloud transitions.

6.4. ShapeNet (Gromov-Wasserstein Wormhole)

Gromov-Wasserstein (GW) (Mémoli, 2011) is an extension of OT which seeks the most isometric map between point clouds. Given two point clouds, their GW distance is:

$$GW(X,Y) = \min_{P \in U(\nu,\mu)} \sum_{i,j,i',j'} \|d^X_{i,i'} - d^Y_{j,j'}\|^2 P_{i,j} P_{i',j'} \tag{11}$$

Where $d^X_{i,i'}$ is the distance between X_i and $X_{i'}$, and similarly for $d^Y_{j,j'}$. This problem is quadratic is significantly more computationally intensive than standard OT (Peyré et al., 2019). Unlike standard OT, GW can align datasets from different modalities (Demetci et al., 2020), map between graphs (Xu et al., 2019) and quantify distances between 3D shapes with arbitrary orientation (Mémoli, 2009).

ShapeNet is a CAD-based dataset of 3D point clouds consisting of 16 different object classes. To construct the conditions where GW is usually applied, we randomly rotated each sample in 3D dimensions (Figure 6a). Standard OT is blind to structure when orientation is unknown, making rotation-invariant GW necessary. We then trained Wormhole to encode the rotated point clouds such that distances between embeddings retain GW distances.

Although the training objective was radically different, Wormhole generalized to GW and learned accurate embeddings (Table 1). In addition, the encodings were reflective of object classes despite input coordinates being randomly rotated, implying that GW Wormhole was able to produce rotation invariant embeddings (Figure 6b).

6.5. Cellular Niches (High-Dimensional Data)

Spatially resolved transcriptomics (ST) are a set of technologies which capture both RNA information and the spatial context of cells to gain insight into tissue organization. OT has been frequently applied to analyze results of ST and unearth the relationship between a cell's phenotype and its

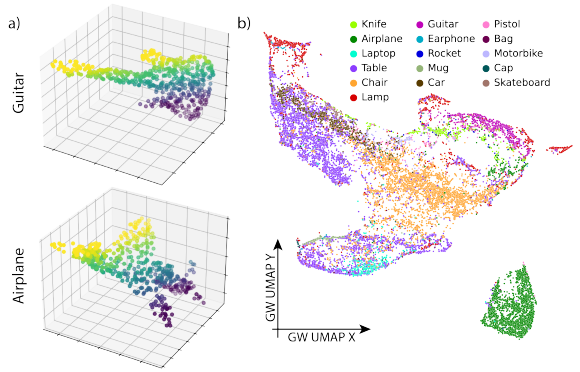

Figure 6. **Gromov-Wasserstein Wormhole. a.** Examples of point clouds from the ShapeNet dataset randomly rotated in 3D to reflect common GW applications. **b.** Embeddings learned by GW Wormhole, visualized in 2D with UMAP and colored by object class. Despite their random orientations, Wormhole was able to encode the ShapeNet point clouds and represent their class, as well as recover GW distances (Table 1).

microenvironment (Cang & Nie, 2020; Haviv et al., 2024). For example, Spatial Optimal Transport (SPOT) (Mani et al., 2022) models the niche of each cell as the point cloud from the gene expression of every cell in its environment, and defines differences between niches using Wasserstein.

While OT is an alluring metric to quantify cellular niches, SPOT cannot scale to ST datasets of typical size. As experimental technologies mature, atlas-level spatial assays continue to grow in ubiquity and size. For example, the 254-gene MERFISH dataset of the motor cortex (Zhang et al., 2021), which consists of 256,058 cells across 64 slices.

We summarize the $50\mu m$ radius around each cell in the MERFISH dataset to construct niche point clouds in 254 dimensions. OT linearization methods such as DWE and DiffusionEMD assume that data is voxelized or discretized. While low-dimensional datasets can be manipulated to fit within their frameworks, the curse of dimensionality prevents adequate voxelization of this *Cell Niches* dataset. As commercial technologies capture more genes (Marco Salas et al., 2023), the need for a method that can scale to higher dimensions, such as Wormhole, becomes more acute.

Parvalbumin-positive (*Pvalb*) interneurons are one the major groups of inhibitory neuron in the cortex, and are crucial for the regulation of neural circuitry (Sohal et al., 2009). They are sparse in the motor cortex, comprising only 3% of the dataset. To understand their spatial organization, it is therefore critical to study the entire cohort and integrate insights from multiple tissue slices.

After training Wormhole on the complete datasets (Table 1), we analyze the embeddings of the niches of *Pvalb* neurons. The first diffusion component (DC) (Coifman & La-

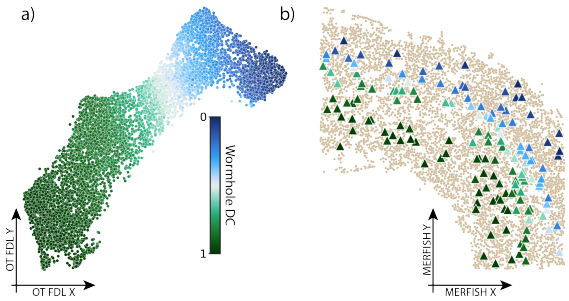

Figure 7. **Wasserstein Wormhole on 254 dimensional spatial transcriptomics data. a.** Visualization of Wormhole embeddings from niches of *Pvalb* neurons from the MERFISH dataset, colored by their first diffusion component. **b.** Spatial organization of *Pvalb* cells in a representative MERFISH slice. Wormhole embeddings reveal the cortical arrangement of these cells, as the first DC corresponds to neuronal depth.

fon, 2006) of Wormhole encodings aligns with their cortical depth across all slices within the dataset (Figure 7a-b). Despite the size and dimensionality of this dataset, Wormhole can still identify key biological signals across the cohort.

7. Discussion

We have introduced Wasserstein Wormhole, an approach for embedding point clouds such that OT distances are preserved and can be approximated efficiently. Although OT is ubiquitous, applying it to cohorts of distributions requires costly computations and scales quadratically with cohort size. Wormhole expands OT to large datasets.

Encoding based on OT distances is theoretically challenged by the fact that Wasserstein is a non-Euclidean metric. To alleviate this concern, we derived theoretical lower and upper bounds that delimit how far non-Euclidean distance matrices can be from the embeddings. We further describe an algorithm based on projected gradient descent which is guaranteed to find an optimal embedding. On examples where the full Wasserstein distance matrix is tractable, Wormhole falls within the bounds and is close to the optimal error.

We applied our approach to eight different settings, ranging from 2D point clouds to massive spatial transcriptomics datasets with hundreds of genes. Across all settings, Wormhole embeddings recapitulated the Wasserstein space, accurately encoded key aspects of the point clouds and outperformed competing approaches. Furthermore, Wormhole's decoder recovered solutions to variational Wasserstein problems such as barycenters and point cloud interpolations. Wormhole seamlessly extends to other OT metrics, such as GW, and learns from arbitrary orientations.

Training Wasserstein Wormhole never exceeded 30 min on a single GPU, even on the 2048 points-per-sample ModelNet40 & ShapeNet datasets. Our algorithm is implemented in JAX (Bradbury et al., 2021) and integrates with OT tools (OTT-JAX) (Cuturi et al., 2022) for efficient Wasserstein distance calculations. Wormhole can be easily applied to any point cloud dataset via a *pip* installable package.

Several methods have been proposed to learn approximations for the Wasserstein distance, including those based on neural networks. Unlike prior methods, however, Wormhole makes no assumptions about empirical distributions and can be applied to general point clouds in arbitrary dimensions, while providing superior performance.

We envision several future directions for this work. Currently, Wormhole is only equipped to handle uniform point-clouds, where each point is equally weighted. Future work can extend the Attention mechanism to include arbitrarily weighted point-clouds, to reflect more general instances of empirical distributions and applications of OT.

Wasserstein is not only used to define a distance between distributions, but also to find the optimal mapping across them. An alluring Wormhole extension would be for it to produce the ideal matching in addition to estimating distance. This could possibly be achieved by exploiting the differentiability of Wormhole embeddings with respect to the point clouds.

Acknowledgements

This work was supported by National Cancer Institute grants P30 CA08748 and U54 CA209975. Dana Pe'er is an Investigator at the Howard Hughes Medical Institute.

Impact Statement

Wasserstein Wormhole is a novel method to scale computations of OT distances to large cohorts. We currently do not foresee any adverse effects of our work

References

Amos, B., Cohen, S., Luise, G., and Redko, I. Meta optimal transport. *arXiv preprint arXiv:2206.05262*, 2022.

Bachmann, F., Hennig, P., and Kobak, D. Wasserstein t-sne. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 104–120. Springer, 2022.

Boyd, S. P. and Vandenberghe, L. *Convex optimization*. Cambridge university press, 2004.

Boyle, J. P. and Dykstra, R. L. A method for finding projections onto the intersection of convex sets in hilbert spaces. In *Advances in Order Restricted Statistical Inference: Proceedings of the Symposium on Order Restricted*

Statistical Inference held in Iowa City, Iowa, September 11–13, 1985, pp. 28–47. Springer, 1986.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., et al. Jax: Autograd and xla. *Astrophysics Source Code Library*, pp. ascl–2111, 2021.

Bunne, C., Stark, S. G., Gut, G., Del Castillo, J. S., Levesque, M., Lehmann, K.-V., Pelkmans, L., Krause, A., and Rätsch, G. Learning single-cell perturbation responses using neural optimal transport. *Nature Methods*, 20(11):1759–1768, 2023.

Cang, Z. and Nie, Q. Inferring spatial and signaling relationships between cells from single cell transcriptomic data. *Nature communications*, 11(1):2084, 2020.

Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

Chewi, S., Clancy, J., Le Gouic, T., Rigollet, P., Stepaniants, G., and Stromme, A. Fast and smooth interpolation on wasserstein space. In *International Conference on Artificial Intelligence and Statistics*, pp. 3061–3069. PMLR, 2021.

Coifman, R. R. and Lafon, S. Diffusion maps. *Applied and computational harmonic analysis*, 21(1):5–30, 2006.

Courty, N., Flamary, R., and Ducoffe, M. Learning wasserstein embeddings. *arXiv preprint arXiv:1710.07457*, 2017.

Cuturi, M. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013.

Cuturi, M. and Doucet, A. Fast computation of wasserstein barycenters. In *International conference on machine learning*, pp. 685–693. PMLR, 2014.

Cuturi, M., Meng-Papaxanthos, L., Tian, Y., Bunne, C., Davis, G., and Teboul, O. Optimal transport tools (ott): A jax toolbox for all things wasserstein. *arXiv preprint arXiv:2201.12324*, 2022.

De Leeuw, J. and Mair, P. Multidimensional scaling using majorization: Smacof in r. *Journal of statistical software*, 31:1–30, 2009.

Delon, J., Desolneux, A., and Salmona, A. Gromov–wasserstein distances between gaussian distributions. *Journal of Applied Probability*, 59(4):1178–1198, 2022.

Demetci, P., Santorella, R., Sandstede, B., Noble, W. S., and Singh, R. Gromov-wasserstein optimal transport to align single-cell multi-omics data. *BioRxiv*, pp. 2020–04, 2020.

Dowson, D. and Landau, B. The fréchet distance between multivariate normal distributions. *Journal of multivariate analysis*, 12(3):450–455, 1982.

Feydy, J., Charlier, B., Vialard, F.-X., and Peyré, G. Optimal transport for diffeomorphic registration. In *Medical Image Computing and Computer Assisted Intervention-MICCAI 2017: 20th International Conference, Quebec City, QC, Canada, September 11-13, 2017, Proceedings, Part I 20*, pp. 291–299. Springer, 2017.

Flamary, R., Courty, N., Gramfort, A., Alaya, M. Z., Boisbunon, A., Chambon, S., Chapel, L., Corenflos, A., Fatras, K., Fournier, N., et al. Pot: Python optimal transport. *The Journal of Machine Learning Research*, 22(1):3571–3578, 2021.

Frogner, C., Mirzazadeh, F., and Solomon, J. Learning embeddings into entropic wasserstein spaces. *arXiv preprint arXiv:1905.03329*, 2019.

Gaffke, N. and Mathar, R. A cyclic projection algorithm via duality. *Metrika*, 36(1):29–54, 1989.

Garcia-Garcia, A., Gomez-Donoso, F., Garcia-Rodriguez, J., Orts-Escolano, S., Cazorla, M., and Azorin-Lopez, J. Pointnet: A 3d convolutional neural network for real-time object class recognition. In *2016 International joint conference on neural networks (IJCNN)*, pp. 1578–1584. IEEE, 2016.

Genevay, A., Peyré, G., and Cuturi, M. Learning generative models with sinkhorn divergences. In *International Conference on Artificial Intelligence and Statistics*, pp. 1608–1617. PMLR, 2018.

Gower, J. Properties of euclidean and non-euclidean distance matrices. *Linear Algebra and its Applications*, 67:81–97, 1985. ISSN 0024-3795. doi: https://doi.org/10.1016/0024-3795(85)90187-9. URL https://www.sciencedirect.com/science/article/pii/0024379585901879.

Guo, Y., Wang, H., Hu, Q., Liu, H., Liu, L., and Bennamoun, M. Deep learning for 3d point clouds: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 43(12):4338–4364, 2020.

Haviv, D., Gatie, M., Hadjantonakis, A.-K., Nawy, T., and Pe'er, D. The covariance environment defines cellular niches for spatial inference. *bioRxiv*, 2023.

Haviv, D., Remšík, J., Gatie, M., Snopkowski, C., Takizawa, M., Pereira, N., Bashkin, J., Jovanovich, S., Nawy, T., Chaligne, R., et al. The covariance environment defines cellular niches for spatial inference. *Nature Biotechnology*, pp. 1–12, 2024.

Higham, N. J. Computing a nearest symmetric positive semidefinite matrix. *Linear algebra and its applications*, 103:103–118, 1988.

Horn, R. A. and Johnson, C. R. *Matrix analysis*. Cambridge university press, 2012.

Katageri, S., Sarkar, S., and Sharma, C. Metric learning for 3d point clouds using optimal transport. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 561–569, 2024.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kruskal, J. B. and Wish, M. *Multidimensional scaling*. Number 11. Sage, 1978.

Lee, J., Lee, Y., Kim, J., Kosiorek, A., Choi, S., and Teh, Y. W. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning*, pp. 3744–3753. PMLR, 2019.

Liu, X., Bai, Y., Lu, Y., Soltoggio, A., and Kolouri, S. Wasserstein task embedding for measuring task similarities. *arXiv preprint arXiv:2208.11726*, 2022.

Lohoff, T., Ghazanfar, S., Missarova, A., Koulena, N., Pierson, N., Griffiths, J., Bardot, E., Eng, C.-H., Tyser, R., Argelaguet, R., et al. Integration of spatial and single-cell transcriptomic data elucidates mouse organogenesis. *Nature biotechnology*, 40(1):74–85, 2022.

Mani, S., Haviv, D., Kunes, R., and Pe'er, D. Spot: Spatial optimal transport for analyzing cellular microenvironments. In *NeurIPS 2022 Workshop on Learning Meaningful Representations of Life*, 2022.

Marco Salas, S., Czarnewski, P., Kuemmerle, L. B., Helgadottir, S., Mattsson Langseth, C., Tiesmeyer, S., Avenel, C., Rehman, H., Tiklova, K., Andersson, A., et al. Optimizing xenium in situ data utility by quality assessment and best practice analysis workflows. *bioRxiv*, pp. 2023–02, 2023.

Mémoli, F. Spectral gromov-wasserstein distances for shape matching. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, pp. 256–263. IEEE, 2009.

Mémoli, F. Gromov–wasserstein distances and the metric approach to object matching. *Foundations of computational mathematics*, 11:417–487, 2011.

Naor, A. and Schechtman, G. Planar earthmover is not in 1_1. *SIAM Journal on Computing*, 37(3):804–826, 2007.

Nitzan, M., Karaiskos, N., Friedman, N., and Rajewsky, N. Gene expression cartography. *Nature*, 576(7785): 132–137, 2019.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

Persad, S., Choo, Z.-N., Dien, C., Sohail, N., Masilionis, I., Chaligné, R., Nawy, T., Brown, C. C., Sharma, R., Pe'er, I., et al. Seacells infers transcriptional and epigenomic cellular states from single-cell genomics data. *Nature Biotechnology*, 41(12):1746–1757, 2023.

Peyré, G., Cuturi, M., et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

Qi, C. R., Su, H., Mo, K., and Guibas, L. J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660, 2017.

Scetbon, M., Cuturi, M., and Peyré, G. Low-rank sinkhorn factorization. In *International Conference on Machine Learning*, pp. 9344–9354. PMLR, 2021.

Schiebinger, G., Shu, J., Tabaka, M., Cleary, B., Subramanian, V., Solomon, A., Gould, J., Liu, S., Lin, S., Berube, P., et al. Optimal-transport analysis of single-cell gene expression identifies developmental trajectories in reprogramming. *Cell*, 176(4):928–943, 2019.

Sehanobish, A., Ravindra, N. G., and van Dijk, D. Permutation invariant networks to learn wasserstein metrics. In *TDA {\&} Beyond*, 2020.

Shirdhonkar, S. and Jacobs, D. W. Approximate earth mover's distance in linear time. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8. IEEE, 2008.

Sohal, V. S., Zhang, F., Yizhar, O., and Deisseroth, K. Parvalbumin neurons and gamma rhythms enhance cortical circuit performance. *Nature*, 459(7247):698–702, 2009.

Solomon, J., De Goes, F., Peyré, G., Cuturi, M., Butscher, A., Nguyen, A., Du, T., and Guibas, L. Convolutional wasserstein distances: Efficient optimal transportation

on geometric domains. *ACM Transactions on Graphics (ToG)*, 34(4):1–11, 2015.

Sonthalia, R., Van Buskirk, G., Raichel, B., and Gilbert, A. How can classical multidimensional scaling go wrong? *Advances in Neural Information Processing Systems*, 34: 12304–12315, 2021.

Stephenson, E., Reynolds, G., Botting, R. A., Calero-Nieto, F. J., Morgan, M. D., Tuong, Z. K., Bach, K., Sungnak, W., Worlock, K. B., Yoshida, M., et al. Single-cell multi-omics analysis of the immune response in covid-19. *Nature medicine*, 27(5):904–916, 2021.

Su, Z., Wang, Y., Shi, R., Zeng, W., Sun, J., Luo, F., and Gu, X. Optimal mass transport for shape matching and comparison. *IEEE transactions on pattern analysis and machine intelligence*, 37(11):2246–2259, 2015.

Tong, A. Y., Huguet, G., Natik, A., MacDonald, K., Kuchroo, M., Coifman, R., Wolf, G., and Krishnaswamy, S. Diffusion earth mover's distance and distribution embeddings. In *International Conference on Machine Learning*, pp. 10336–10346. PMLR, 2021.

Torgerson, W. S. Multidimensional scaling: I. theory and method. *Psychometrika*, 17(4):401–419, 1952.

Torrence, C. and Compo, G. P. A practical guide to wavelet analysis. *Bulletin of the American Meteorological society*, 79(1):61–78, 1998.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Villani, C. et al. *Optimal transport: old and new*, volume 338. Springer, 2009.

Wu, F., Courty, N., Jin, S., and Li, S. Z. Improving molecular representation learning with metric learning-enhanced optimal transport. *Patterns*, 4(4), 2023.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1912–1920, 2015.

Xu, H., Luo, D., Zha, H., and Duke, L. C. Gromov-wasserstein learning for graph matching and node embedding. In *International conference on machine learning*, pp. 6932–6941. PMLR, 2019.

Zhang, M., Eichhorn, S. W., Zingg, B., Yao, Z., Cotter, K., Zeng, H., Dong, H., and Zhuang, X. Spatially resolved cell atlas of the mouse primary motor cortex by merfish. *Nature*, 598(7879):137–143, 2021.

Wasserstein Wormhole - Supplementary Materials

8. Permutation Equivariance of Attention

The permutation equivariance feature of self-attention is a critical component of our Wormhole algorithm. This means that switching the order of points similarly switches the order of the Attention encodings. While generally known (Lee et al., 2019), due to its relevance in our work, we provide a proof of the equivariance here too. Given weights $W_K, W_V, W_Q \in R^{d\times d}$, the attention operator on input $X \in R^{n\times d}$ is:

$$A(X) = \text{softmax}\left(\frac{XW_QW_K^\top X^\top}{\sqrt{d}}\right)XW_V \tag{12}$$

So when rows of X are permuted via matrix P, the attention output is:

$$A(PX) = \text{softmax}\left(\frac{PXW_QW_K^\top X^\top P^\top}{\sqrt{d}}\right)PXW_V \tag{13}$$

Since *softmax* is an element wise operation, we can factor out the permutation. From the unitary property of permutation matrices ($P^\top = P^{-1}$):

$$A(PX) = P\cdot\text{softmax}\left(\frac{XW_QW_K^\top X^\top}{\sqrt{d}}\right)XW_V \tag{14}$$

Which shows that attention operator is permutation equivariant.

9. Bounds on Embedding non-Euclidean Distances

Let $D \in \mathbb{R}^{N\times N}$ by a pairwise distance matrix. Following definition 3.1, if D is Euclidean (EDM), is can be perfectly represented as the pairwise distances between n points through multi-dimensional scaling (MDS) (Kruskal & Wish, 1978). Since the Wasserstein distance is non-Euclidean (Peyré et al., 2019), we are interested in how well we can approximate non-Euclidean distance matrices with pairwise distances between learned embeddings.

Here, we consider the problem of finding the closet valid EDM to D:

$$S = \min_{D'} \|D - D'\|_F^2 \tag{15}$$

with the constraints reflecting D' being an EDM:

$$C_1\text{: } -J\hat{D}J \text{ is PSD, } C_2\text{: } \hat{D} \text{ is hollow and } C_3\text{: } \hat{D}_{i,j} \geq 0.$$

This problem is fully convex and has a global minimum which we call $\boldsymbol{\mathcal{L}}^*$. In the following, we derive a lower and upper for $\boldsymbol{\mathcal{L}}^*$. To find a lower bound, we solve a simplified problem by eschewing the hollowness and non-negativity constraints on D'. Then, we will reintroduce those constraints and use the solution for the simplified problem to construct an upper bound.

9.1. Error Lower Bound

Let $P = \frac{1}{N}\mathbf{1}_N\mathbf{1}_N^\top$. We can write the identity matrix as $I = J + P$ and turn our objective into:

$$\|D - D'\|_F^2 = \|(J+P)D(J+P) - (J+P)D'(J+P)\|_F^2 \tag{16}$$

Expanding the products:

$$\|D - D'\|_F^2 = \|A + B\|_F^2 \tag{17}$$

where $A := JDJ - JD'J$ and $B := PD'J + JD'P + PD'P - PDJ - JDP - PDP$. From the properties of the Frobenius norm:

$$\|A + B\|_F^2 = \|A\|_F^2 + \|B\|_F^2 + 2Tr(A^\top B) \tag{18}$$

Since the values of P are constant $J \cdot P = P \cdot J = 0$, and therefore the only non-zero term in AB are:

$$AB = J(D - D')J(D - D')P \tag{19}$$

Trace of matrix product is invariant under cyclic permutation, which means:

$$Tr(AB) = Tr(J(D - D')J(D - D')P) = 0 \tag{20}$$

where again we use the identity that $P \cdot J = 0$. Plugging that back in equation 18:

$$\|D - D'\|_F^2 = \|A\|_F^2 + \|B\|_F^2 \tag{21}$$

Conceptually, the $\|A\|_F^2$ term is optimized when the D' and D are equal after centering, and therefore does not depend on their row/columns means. Meanwhile the $\|B\|_F^2$ term is optimized when the row/columns means of D' and D are equal.

When only the constraint on $F' = -JD'J$ being PSD is maintained, the two terms $\|A\|_F^2$ and $\|B\|_F^2$ can be solved independently to derive an analytical solution. The simplified version of optimizing $\|A\|_F^2$ is:

$$\begin{aligned} &\min_{D_A} \|JD_AJ - JDJ\|_F^2 \\ &\text{s.t.} - JD_AJ \text{ is PSD} \end{aligned} \tag{22}$$

This describes the problem of finding the closest PSD approximation to a symmetric matrix (Higham, 1988), whose solution is truncation of the negative eigenspace of JDJ:

$$D_A = -\sum_{i:\lambda_i > 0} \lambda_i v_i v_i^\top \tag{23}$$

where $\{\lambda_i, v_i\}_{i=1}^N$ are the eigenvalues and eigenvectors of $F = -JDJ$.

Since JDJ has row/column means 0, the ones vector $\mathbf{1}_n$ is an eigenvector of JDJ with eigenvalue 0. Therefore $\mathbf{1}_n$ is also in the null space D_A and:

$$D_A = JD_AJ \tag{24}$$

and

$$PD_A = D_A P = 0 \tag{25}$$

To solve $\|B\|_F^2$, we apply the definition of J and arrive at:

$$PDJ + JDP + PDP = PD + (PD)^\top - PDP \tag{26}$$

We denote the mean of the i-th row/column of D as $\hat{D}_i$ and the mean of all elements as $\hat{D}$. From the definition of P, we can set:

$$D_B^{i,j} = (PDJ + JDP + PIP)_{i,j} = \hat{D}_i + \hat{D}_j - \hat{D} \tag{27}$$

The row/columns mean of D_B equal those of D itself, and therefore D_B solves $\|B\|_F^2$ with zero error. Since $JD_BJ = 0$, $D' = D_A + D_B$ is a minimizes both $\|A\|_F^2$ and $\|B\|_F^2$:

$$\begin{gathered} D_A + D_B = \arg\min_{D'} \|D - D'\|_F^2 \\ \text{s.t.} - JD'J \text{ is PSD} \end{gathered} \tag{28}$$

With an error of:

$$\mathbf{L} = \|D - (D_A + D_B)\|_F^2 = \sum_{i:\lambda_i<0} \lambda_i^2 \tag{29}$$

Because equation 28 has only a single constraint out of the 3 required from Euclidean matrices, $\mathbf{L}$ is a lower bound for the complete problem $\boldsymbol{\mathcal{L}}^* \geq \mathbf{L}$.

9.2. Error Upper Bound

The truncation step at equation 23 changes the diagonal elements and $D_A + D_B$ (equation 28) is strictly not hollow nor non-negative when D is not Euclidean. However, we can consider what can be added to D_A such the resulting matrix is hollow and non-negative but without incurring additional penalty on $\|A\|_F^2$. To this end, we will make use of the following lemma:

Lemma 9.1. *For a given symmetric matrix A, there is unique symmetric matrix B such that $A + B$ is hollow and $JBJ = 0$.*

Proof. From the constraint:

$$JBJ = 0 \Longrightarrow J(BJ) = (JB)J = 0 \tag{30}$$

That is that BJ is in the right null space of J and JB is in the left null space of J. The former implies that the rows of BJ are constant and can be written as:

$$BJ = B(I - P) = \mathbf{1}_N \cdot c^\top \Longrightarrow B = BP + \mathbf{1}_N \cdot c^\top \tag{31}$$

Where c is a columns vector. Multiplying equation 31 on the left by P we get:

$$PB = PBP + P\mathbf{1}_N \cdot c^\top = \hat{B} + \mathbf{1}_N \cdot c^\top \tag{32}$$

Where $\hat{B}$ is a matrix filled with the mean value of B. Which in turn implies:

$$c^\top = \frac{1}{N}\mathbf{1}_N^\top B - \hat{B}\mathbf{1}_N^\top \tag{33}$$

So by the symmetry of B we arrive at:

$$B = \mathbf{1}_N \frac{\mathbf{1}^\top B}{N} + \frac{B\mathbf{1}_N}{N}\mathbf{1}_N^\top - \hat{B} \tag{34}$$

$B\mathbf{1}$ is a column vector, so to fit this constraint, B must be in the form of:

$$B = \mathbf{1}_N \frac{b^\top}{N} + \frac{b}{N}\mathbf{1}_N^\top - \hat{b}\frac{\mathbf{1}_N\mathbf{1}_N^\top}{N} \tag{35}$$

Therefore, we have only n degrees of freedom with which to find B. However, since $A + B$ has diagonal 0, this produces a series of N equations which the diagonal elements of B must fit:

$$b_i - \frac{1}{2}\hat{b} = -\frac{1}{2}A_{i,i} \tag{36}$$

The matrix describing this linear system has ones along its diagonals while its top triangular section is filled it $-\frac{1}{2\cdot N}$. Such a system is well-defined and therefore ascribes a unique solution onto b.

□

According to lemma 9.1, for the matrix $D_A = -\sum_i \lambda_i \geq 0 \lambda_i v_i v_i^\top$ there is a unique matrix $G = g \cdot 1^\top + 1 \cdot g^\top - \hat{g}$ such that $D_A + G$ is hollow and $JGJ = 0$. From the construction, we find that the diagonal elements of D_A are:

$$D_A^{i,i} = -\sum_{j:\lambda_j > 0} \lambda_j \cdot v_{i,j}^2 \tag{37}$$

So by solving equation 36:

$$\begin{aligned} g_i &= \frac{1}{2}\Big(\sum_{j:\lambda_j>0} \lambda_j \cdot v_{i,j}^2 + \frac{1}{N}\hat{g}\Big) \\ \hat{g} &= \frac{1}{n}\sum g_i = \frac{1}{N}\sum_i \sum_{j:\lambda_j>0} \lambda_j \cdot v_{i,j}^2 \end{aligned} \tag{38}$$

We get that the matrix G is:

$$G_{i,j} = -\frac{D_A^{i,i} + D_A^{j,j}}{2} \tag{39}$$

While lemma 9.1 assists us with conforming to the hollowness constraints, we must also adhere to non-negativity. To this end, we will also apply the lemma shown below:

Lemma 9.2. *For a negative-semi-definite (NSD) matrix M, the matrix $Q = M + C$ where:*

$$C_{i,j} = -\frac{M_{i,i} + M_{j,j}}{2} \tag{40}$$

Has strictly non-negative values:

$$Q_{i,j} \geq 0 \tag{41}$$

Proof. The matrix $W = -M$ is PSD, and so is every 2×2 principle submatrix of $-M$ which implies (Horn & Johnson, 2012):

$$W_{i,j} \leq \sqrt{W_{i,i} W_{j,j}} \tag{42}$$

From the AM-GM inequality:

$$W_{i,j} \leq \frac{W_{i,i} + W_{j,j}}{2} \tag{43}$$

And since $W_{i,j} = -M_{i,j}$ we arrive at:

$$M_{i,j} \geq \frac{M_{i,i} + M_{j,j}}{2} \tag{44}$$

And finally:

$$Q_{i,j} = M_{i,j} + C_{i,j} = M_{i,j} - \frac{M_{i,i} + M_{j,j}}{2} \geq 0 \tag{45}$$

□

From lemma 9.2, the matrix $D_A + G$ fits all 3 constraints, since it is also hollow, non-negative and $F = -J(D_A + G)J = -JD_AJ$ is PSD.

While the resulting G matrix does not change the value of $\|A\|_F^2$, it differs from D_B and incurs an error in the $\|B\|_F^2$ term. The sum of the untruncated version of JDJ and the matrix $PDJ + JDP + PDP$ equals to D (equation 16) and is hollow. From lemma 9.1, the latter matrix can be expressed similarly to equation 38 by including the negative eigenvalues:

$$\begin{aligned} \tilde{g}_i &= \frac{1}{2}(\sum_j \lambda_j \cdot v_{i,j}^2 + \frac{1}{N}\tilde{\hat{g}}) \\ \tilde{\hat{g}} &= \frac{1}{N}\sum g_i = \frac{1}{N}\sum_i \sum_j \lambda_j \cdot v_{i,j}^2 \end{aligned} \tag{46}$$

So if we define:

$$\Delta g_i = \frac{1}{2} \sum_{j:\lambda_j<0} \lambda_j \cdot v_{i,j}^2 \tag{47}$$

We get that the incurred error in the $\|B\|_F^2$ term is:

$$\|B\|_F^2 = \|\tilde{G} - G\|_F^2 = \sum_{i,j} (\Delta g_i + \Delta g_j)^2 \tag{48}$$

Which, included with the truncation error $\|A\|_F^2$, results in an upper bound to the complete problem:

$$\boldsymbol{\mathcal{L}}^* \leq \mathbf{U} = \sum_{i,j} (\Delta g_i + \Delta g_j)^2 + \sum_{j:\lambda_j<0} \lambda_j^2 \tag{49}$$

10. Guaranteed Optimal Embeddings for non-Euclidean Distances

As formulated in equation 15, the problem of finding the closest valid EDM to an arbitrary distance matrix involves minimizing a convex function subject to convex constraints. Formally, the constraint of D' being an EDM is the intersection of three convex sets, and therefore, projected gradient descent (Boyd & Vandenberghe, 2004) (PGD) is guaranteed to converge to the global minimum $\boldsymbol{\mathcal{L}}^*$:

- (C_1) Matrices D' such that $F = -JD'J$ is PSD
- (C_2) Hollow Matrices $D'_{i,i} = 0 \; \forall i$
- (C_3) Non-Negative Matrices $D'_{i,j} \geq 0 \; \forall i, j$

Starting from some initial value D'_0, each gradient descent step is defined as:

$$D'_{t+1} = \text{proj}_{C_1 \cup C_2 \cup C_3}(D'_t - \varepsilon_t \nabla S(D'_t)) \tag{50}$$

Where S is the error term in equation 15 and ε_t is the step size at iteration t. In subsection 9.1, we have demonstrated how to project a matrix K onto the set of matrices where $-JKJ$ is PSD:

$$K_{C_1} = \text{proj}_{C_1}(K) = M + PK + (PK)^\top + PKP \tag{51}$$

Where M is the truncation of $-JKJ$ to its positive eigenspace:

$$M = -\sum_{i:\lambda_i > 0} \lambda_i v_i v_i^\top \tag{52}$$

And $\{\lambda_i, v_i\}_{i=1}^N$ are the eigenvectors and values of $F = -JKJ$. Projecting a matrix K onto the set of hollow matrices is simply stetting the diagonal elements to 0:

$$K_{C_2} = \text{proj}_{C_2}(K) = K - \text{Diag}(K) \tag{53}$$

And so is projecting onto non-negative matrices straightforward thresholding:

$$K_{C_3} = \text{proj}_{C_3}(K) = \begin{cases} K_{i,j} & \text{if } K_{i,j} \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{54}$$

To project onto the intersection of C_1, C_2 and C_3 we can use Dykstra's algorithm (Boyle & Dykstra, 1986; Gaffke & Mathar, 1989), which iterativly projects onto each convex set using corrective variables to find the element closest to K in $C_1 \cup C_2 \cup C_3$.

Using projected gradient descent (PGD) as described in equation 50 we are sure to find the closest valid EDM to an arbitrary distance matrix D (Algorithm 2). We can then apply classical MDS to produce embeddings whose distance match the idea euclidean matrix D'. The error of this solution is guaranteed to be within the lower and upper bounds derived in subsections 9.1 and 9.2.

Despite its theoretical soundness, there are hurdles preventing this algorithm from being applied on real-world datasets. It relies on having access to the complete distance matrix D, which not trivial for large datasets and complex metrics such as the Wasserstein Distance. This algorithm is indeed infeasible on large cohorts, which motivates our stochastic Wormhole approach.

10.1. Theoretical Results

On small examples, we can however implement algorithm 2 and empirically show that its solution is within our derived error bounds. We construct an a cohort of empirical distribution whose pairwise Wasserstein distance is non-Euclidean

based on the example presented in section 8.3 in (Peyré et al., 2019). We start by generating a set of distributions in R^2 with support on $x_1 = (0, 0), x_2 = (1, 0), x_3 = (1, 0), x_4 = (1, 1)$ where the probability of each points is either $\{0, \frac{1}{4}, \frac{2}{4}, \frac{3}{4}, 1\}$ such that the weights sum to 1 for each distribution, which produces $N = 35$ weighted point clouds.

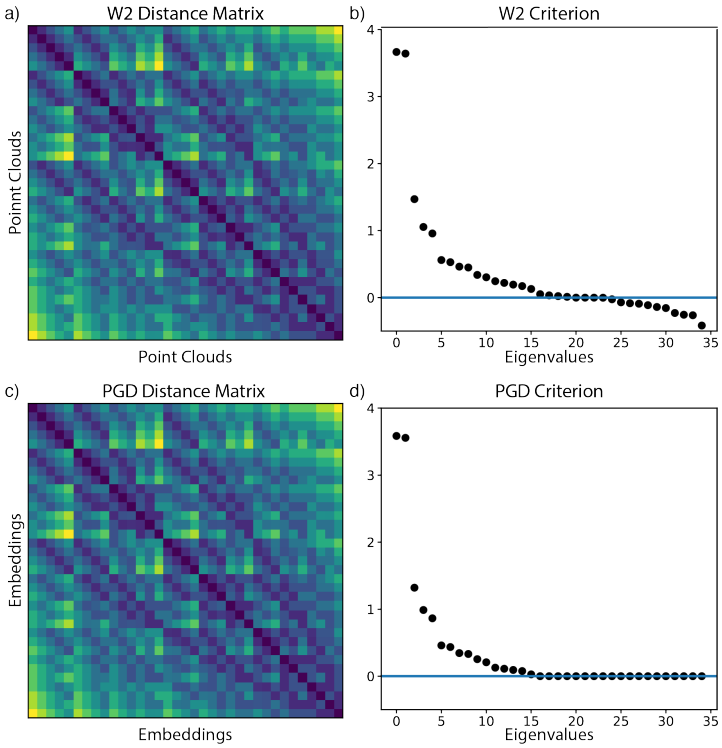

Figure S1. **Non-Euclidean Wasserstein distance matrix and PGD convergence to it. a.** Pairwise Wasserstein distances between the 35 *Simplex* point clouds. **b.** Spectrum of the double centered Wasserstein distance matrix (i.e. $F = -JDJ$). Since there are negative eigenvalues, the original distance matrix is non-Euclidean. **c.** Distance matrix produced PGD algorithm 2 to fit a valid EDM. **d.** Spectrum of the double centered PGD distance matrix. The PGD distance matrix is an EDM since it is hollow, positive and its criterion matrix is PSD.

To fit within our framework of empirical distributions, we convert each weighted point cloud to a uniform one by duplicating each point based on its probability. For example, if a point is weighted $\frac{3}{4}$, we duplicate it so that it appears thrice with uniform weight. We then add a small amount of independent Gaussian noise to make the points in each cloud are all different from one another. This scheme produces a set of 35 point clouds each with 4 points whose pairwise Sinkhorn divergence distance matrix is non-Euclidean (Figure S1a-b). Since original point weights are derived from the 4 dimensional simplex, we dub this the *Simplex* example.

We apply Algorithm 2 on the *Simplex* distance matrix and find that the resulting error fits well within our derived bounds (Table 2 and Figure S1c-d). We similarly apply our Wormhole algorithm to produce embeddings and classical MDS with embedding dimension of 35 (Table 2). Despite the stochastic training of Wormhole, its error is close to the global minima derived by PGD. As expected, cMDS performs poorly and its error is outside of the upper bound.

We compare Wormhole to PGD and cMDS on two more non-Euclidean examples. The first, *Gaussians*, we generate by randomly producing $N = 128$ mean vectors (μ) of length $d = 8$ and N covariance matrices (Σ) of shape $d \times d$. Both μ vectors and Σ matrices are sampled from the standard normal, where Σ matrices are ensured to valid covariances (i.e. PSD) by multiplying resulting matrices with their transpose. From each of the N resulting parameterized Gaussians we randomly samples 512 points to create point clouds. Values are then normalized to be between -3 and 3. The pairwise Sinkhorn divergence between these point clouds is not Euclidean as the minimum eigenvalue of the resulting criterion matrix was $\lambda_{min} = -0.051$.

An important insight from our upper and lower bounds derived in subsections 9.1 and 9.2 is that our Wormhole algorithm is expect to perform well in cases where the negative eigenspace is considerably smaller than its positive counterpart. To demonstration of the non-Euclidean nature of the Wasserstein distance in our real-world datasets and explain the accuracy of our approximation nevertheless, we draw $N = 256$ point clouds from the binarized *MNIST* dataset. The pairwise Sinkhorn divergence matrix was non-Euclidean, and its criterion matrix did have negative eigenvalues $\lambda_{min} = -0.062$. However, unlike the *Simplex* example, on this sample of real dataset the negative eigenspace was much smaller in magnitude and less prominent (Figure S2).

On both the *Gaussian* and *MNIST* examples, we find that both Wormhole and PGD error terms are within the lower and upper error bounds (Table 2). We again stress that the PGD algorithm 2 is not realizable on large datasets, and the stochastic nature of Wormhole enables its scalability, albeit at a marginal cost compare to optimal solution.

11. Implementation Details

We made use of three different Wasserstein approximation algorithms in our paper, DiffusionEMD, Fréchet distances and our novel Wormhole. In the following, we describe how each approach was applied.

11.1. Wasserstein Wormhole

11.1.1. Pre-processing

Stability and efficiency of Wasserstein distance computations, specifically entropically regularized OT, is highly dependent on the scale of the input. The larger the values in the ground-cost distance matrix, the more iterations of Sinkhorn are required and the likelihood of machine-point precision error (*nan*) grows.

To mitigate *nan*-related risks and to ensure swift training of Wormhole, by default we scale the point cloud values to reside between -1 and 1 via min-max normalization. For high-dimensional distributions, such as the *Cellular Niche* cohort in subsection 6.5, we further scale down by a factor of square-root number of dimensions. To accommodate instances where no re-scaling is desired, this operation can be easily disabled by the user.

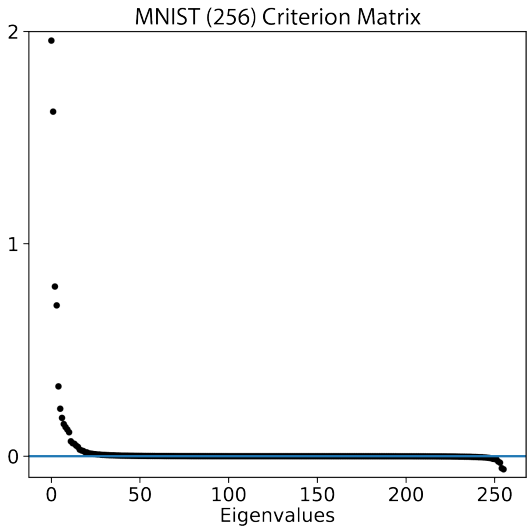

Figure S2. **Spectrum of the criterion matrix from real data**. On the randomly sampled *MNIST*, we calculated the 256×256 pairwise Wasserstein distance matrix and the eigenvalues of its criterion matrix. While D is not Euclidean, as $F = -JDJ$ has negative eigenvalues, their contribution to the spectrum is not as prominent relative to the *Simplex* case (Figure S1).

11.1.2. Neural Architecture

Wasserstein Wormhole uses Transformers to both encode point clouds into embeddings, and reconstruct them from the embedding using the decoder. By default, the encoder consists of an *embedding* layer to transform the coordinate into the dimension of the encoding. This is followed by 3 attention blocks which consist of 4-headed attention followed by a 2-layer fully connected (FC) network. The embedding dimension is set to be 128 and the size of the hidden layer in FC network within the attention block is 512 neurons. To produce the permutation-invariant embedding, we average the attention encodings of each sample within the point cloud.

The decoder only has purview onto the embedding, and thus lacks explicit knowledge of the number of samples within the original empirical distribution. Since the decoder is trained via Wasserstein distance with respect to the input point cloud, there is not constraint on decoder to produce an output with the same number of points. For simplicity, we set the decoder to produce a empirical distribution with m points, where m is the median sample size of cohort, as noted in Table 1.

Given an encoding, the decoder first applies a linear, *multiplier* layer, which returns m new embeddings, one for each point in the output. From there, the default decoder architecture mirrors the encoder, with 3 attention blocks followed by an *unembedding* layer to project back into the original point cloud dimension.

11.1.3. Training and OT Parameters

Wormhole is trained with $10,000$ gradient descent (GD) steps using the ADAM optimizer (Kingma & Ba, 2014) with an initial learning rate of 10^{-4} and an exponential decay schedule. The default batch size was set to 16, which might seem small, but requires $\binom{16}{2} = 120$ computations of Wasserstein distance in each iteration step to compute the encoding loss. With the default settings, as long as the cohort consists of more than 1600 point clouds, Wormhole requires fewer OT computations to derive embeddings compared to direct pairwise Wasserstein.

Convergence speed of Sinkhorn iteration depends to the level of entropic regularization, with larger values of ε (equation 1) resulting in quicker convergence. Within Wormhole, we have two choices for ε, one for the encoder and another for the decoder. Since the former loss requires many more computations, its default is $\varepsilon_{enc} = 0.1$, while we set the regularization at the decoder to a smaller value of $\varepsilon_{dec} = 0.01$.

In all experiments of Wormhole, we used the Sinkhorn divergence loss (equation 2) instead of standard entropic Wasserstein (equation 1). This includes the GW examples, where we used its Sinkhorn divergence equivalent. These choice reflects our desire that similar point-clouds have close embeddings, rather than their distances reflecting self-transport cost. We additionally implemented other OT distances which users of Wormhole can choose from. Namely, we included the standard (entropic) Wasserstein and GW distances, and also EMD (W_1) and its Sinkhorn divergence S_1, which use L_1 ground-distance instead of L_2.

To calculate the object class accuracy of learned Wormhole embeddings, we train an MLP classifier using the default implementation from scikit-learn (Pedregosa et al., 2011), on the train-set encodings and labels. We opt to a neural network instead of other alternatives such as kNN or Random-Forest due to the dimensionality of the embeddings. We report accuracy of the classifier's predictions the test-set encodings and their true labels.

11.2. DiffusionEMD

The key concept behind DiffusionEMD is the Wavelet approximation to EMD. Briefly, (Shirdhonkar & Jacobs, 2008) showed that the Euclidean distance between Wavelet (Torrence & Compo, 1998) decomposition of an 2D voxelized histograms (i.e. gray-scale image) is correlated with their W_1 distance. By applying the spectral analogous of Wavelets, DiffusionEMD extends (Shirdhonkar & Jacobs, 2008) from images to graphs, and generalize it to more than 2 dimensions and requiring fewer constrains on the ground space.

Applying DiffusionEMD requires first constructing a graph of the support of the distribution. For images, such as MNIST and FashionMNIST, this meant building a graph with $28 \times 28 = 784$ nodes, one for each pixel, and edges are connected nodes if distance between their pixels is smaller than $\sqrt{2}$. For 3D point clouds from ModelNet40 and ShapeNet, we first divided their spaces into $16^3 = 256$ voxels, and nodes were connected in the graph if they are closer than $\sqrt{3}$ units away.

Dataset	Wormhole	DWE
MNIST	$\mathbf{2.57 \cdot 10^{-5}}$	$1.63 \cdot 10^{-4}$
FashionMNIST	$\mathbf{1.41 \cdot 10^{-5}}$	$1.88 \cdot 10^{-4}$
ModelNet40	$\mathbf{8.89 \cdot 10^{-5}}$	$3.52 \cdot 10^{-4}$
ShapeNet	$\mathbf{1.14 \cdot 10^{-4}}$	$6.66 \cdot 10^{-4}$
Rotated ShapeNet (GW)	$\mathbf{2.28 \cdot 10^{-3}}$	NA
MERFISH Cell Niches	$\mathbf{6.94 \cdot 10^{-3}}$	OOM
SeqFISH Cell Niches	$\mathbf{9.90 \cdot 10^{-4}}$	OOM
scRNA-seq Atlas	$\mathbf{3.80 \cdot 10^{-5}}$	OOM

Table 3. Test-set Mean Squared Error (MSE) measuring the discrepancy between embedding distances and Wasserstein distances. Values are derived by 10 random instances of sampling 128 point clouds for each dataset and MSE is computed between the embedding and OT pairwise distance matrices. Since DiffusionEMD only produces embedding which are only correlated OT, it was not included in this benchmarking. Similarly, Fréchet is designed to approximate the unregularized OT, while Wormhole and DWE both are trained on entropic OT, so Fréchet was excluded as well. NA, Not Applicable. OOM, Out-Of-Memory.

Each point cloud is then considered a distribution over the graph, which is based on the number of points within each voxel or grid point. We applied DiffusionEMD with default parameters following instruction in `github.com/KrishnaswamyLab/DiffusionEMD`. DiffusionEMD calculates the spectral wavelet decomposition which acts as its OT preserving embeddings. Object classification is based on a kNN ($k = 1$) classifier between train and test-set embeddings, repeating the process from the original manuscript.

This approach can be applied for general point clouds, and does not necessarily require voxelization. However, since every single point among all point clouds would then require its own node in the graph, DiffusionEMD will not scale in that case. This prevented us from applying it on the two high-dimensional *Cell Niches* and scRNA-seq datasets, and DiffusionEMD produces an out-of-memory (OOM) errors.

11.3. DWE

Deep Wasserstein Embeddings (DWE) (Courty et al., 2017) is a CNN based embedding algorithm for gray-scale images. Conceptually, DWE uses a Siamese-training scheme, where pairs of images are encoded simultaneously by a convolutional auto-encoder, and the network is trained so that distances between embeddings match the Wasserstein Distances between the images. A decoder is also trained to minimize the KL divergence between output and input images.

This generates a parametric embedding scheme that reduces the computational complexity for Wasserstein based analysis. However, the CNN backbone limits the applicability of DWE, and it can only embed 2D images, or point-clouds whose support is a fixed-grid. Indeed, DWE cannot embed cohorts of point-clouds in arbitrary space and high-dimensions which cannot be voxelized.

The original code-base for DWE is written in *python2*, and is unrunnable on modern machines. To efficiently benchmark DWE against Wormhole and other Wasserstein acceleration approaches, we re-wrote the DWE code in *python3* and JAX `github.com/DoronHav/jax_dwe`. We also extended DWE from 2D images to 3D grid by adding 3D convolutional layers. MNIST and FashionMNIST, already being 2D gray-scale images, did not require any processing for DWE to be applied. Being 3D point-clouds, the ShapeNet and ModelNet40 cohorts were first voxelized into a $16^3 = 256$ grid. All other datasets were too high-dimensional for DWE and voxelization produced OOM errors.

The optimization of DWE followed the same procedure as Wormhole, where the model was trained for 10,000 gradient descent steps with the ADAM optimizer and an exponential decay schedule. To match the per-step Wasserstein comparisons of Wormhole, the batch size for DWE was 128. Like Wormhole, DWE embeddings were trained to reconstruct Sinkhorn divergence to allow for direct comparisons between the algorithms. Despite similar training parameters, Wormhole recovered OT space more reliably than DWE on low-dimensional datasets (Tables 1, 3).

11.4. Fréchet

Acting as a naive baseline, this approach utilizes the closed form solution OT has in the Gaussian case as an approximation for general point clouds. For all empirical distribution in a cohort, we calculate their d-dimensional mean and $d \times d$ covariance matrix. We then compute the Fréchet distance (equation 10) between each pair to estimate their Wasserstein distance.

The cost of each Fréchet computation is $O(d^3)$. As long as $d \ll m$, where m is the number of points in each cloud, Fréchet is significantly less computationally intensive than Sinkhorn iterations. On high dimensional datasets, $d \not\ll m$, and Fréchet can not be applied.

This approach does not provide us with an embedding. To still assign test-set with label predictions, we manually perform a kNN ($k = 30$) classifier based on Fréchet distances. We calculate the approximated distance for each test-set point cloud to every train-set example and assign it the most common label among its k closet neighbours.

11.4.1. Gaussian Gromov-Wasserstein

For GW, the formula written in equation 10 no longer applies. While a closed form solution for GW in the Gaussian case has not been found, (Delon et al., 2022) derived a lower bound to it, which we used in lieu. For two Gaussians $N(\mu_1, \Sigma_1)$ and $N(\mu_2, \Sigma_2)$ in d dimensions, their GW distance is greater or equal to:

$$\begin{aligned} LGW^2 = 4(\text{Tr}(D_1) - \text{Tr}(D_2))^2 + 4(\|D_1\|_F - \|D_2\|_F)^2 \\ + 4(\|D_1\|_F^2 - \|D_2\|_F^2) + 4(\|D_1 - D_2\|_F^2) \end{aligned} \tag{55}$$

Where D_i is the diagonalization of Σ_i, sorted in decreasing order. Using equation 55, we estimate pairwise GW distances between Rotated ShapeNet point cloud, which we assign test-set labels using the same kNN classification scheme described in subsection 11.4