A Training algorithm

Algorithm 1 End-to-end training of GeONet

```
Input: data pairs $(\mu_0^{(1)}\mu_1^{(1)}), \dots, (\mu_0^{(n)}, \mu_1^{(n)})$; batch size $N$; initialization of the neural network parameters $\phi, \psi \in \Theta \times \Theta \times \Xi$; weight parameters $\alpha_1, \alpha_2, \beta_0, \beta_1$; domain $\Omega$ and branch domain (mesh) $\tilde{\Omega}$.; denote $i \in \{1, \dots, N\}$.
1: while $\mathcal{L}_{\text{total}}$ has not converged do
2:     Independently draw $N$ sample points from $U(\Omega) \times U(0,1)$, $N$ points from $U(\tilde{\Omega})$, and $N$ density pairs from $\{(\mu_0^{(\ell)}, \mu_1^{(\ell)})\}_{\ell=1}^n$, possibly repeating.
3:     Compute $\Phi_i = \partial_t \mathcal{C}_{\phi,i} + \text{div}(\mathcal{C}_{\phi,i} \nabla \mathcal{H}_{\psi,i})$.    ▷ continuity residual
4:     Compute $\Psi_i = \partial_t \mathcal{H}_{\psi,i} + \frac{1}{2}\|\nabla \mathcal{H}_{\psi,i}\|_2^2$.    ▷ Hamilton-Jacobi residual
5:     Compute $B_{0,i} = \mathcal{C}_{\phi,0,i} - \mu_0^{(i)}(x_{\tilde{\Omega}}^i)$, $B_{1,i} = \mathcal{C}_{\phi,1,i} - \mu_1^{(i)}(x_{\tilde{\Omega}}^i)$.  ▷ boundary residual
6:     Compute
           $\mathcal{L}_{\text{cty}} = \frac{\alpha_1}{N}\sum_{i=1}^N \Phi_i^2, \quad \mathcal{L}_{\text{HJ}} = \frac{\alpha_2}{N}\sum_{i=1}^N \Psi_i^2,$
           $\mathcal{L}_{\text{BC}} = \frac{1}{N}\sum_{i=1}^N (\beta_0 B_{0,i}^2 + \beta_1 B_{1,i}^2),$
7:     Compute $\mathcal{L}_{\text{total}}(\phi,\psi) = \mathcal{L}_{\text{cty}} + \mathcal{L}_{\text{HJ}} + \mathcal{L}_{\text{BC}}$.
8:     Minimize $\mathcal{L}_{\text{total}}(\phi,\psi)$ to update $\phi$ and $\psi$.    ▷ minimize the loss function
9: end while
```

B Derivation of primal-dual optimality conditions for dynamical OT problem

The primal-dual analysis is a standard technique in the optimization literature such as in analyzing certain semidefinite programs (Chen and Yang, 2021). Recall the Benamou-Brenier fluid dynamics formulation of the static optimal transport problem

$$\min_{(\mu,\mathbf{v})} \int_0^1 \int_{\mathbb{R}^d} \frac{1}{2}||\mathbf{v}(x,t)||_2^2 \, \mu(x,t) \, \mathrm{d}x \, \mathrm{d}t \tag{18}$$

$$\text{subject to } \partial_t \mu + \text{div}(\mu \mathbf{v}) = 0, \tag{19}$$

$$\mu(\cdot,0) = \mu_0, \ \mu(\cdot,1) = \mu_1. \tag{20}$$

Here, equation (19) is referred to as the *continuity equation* (CE), preserving the unit mass of the density flow $\mu_t = \mu(\cdot,t)$. We write the Lagrangian function for any flow $(\mu_t)_{t\in[0,1]}$ initializing from μ_0 and terminating at μ_1 as

$$L(\mu,\mathbf{v},u) = \int_0^1 \int_{\mathbb{R}^d} \left[\frac{1}{2}\|\mathbf{v}\|_2^2 \mu + (\partial_t \mu + \text{div}(\mu\mathbf{v}))\, u\right] \mathrm{d}x \, \mathrm{d}t, \tag{21}$$

where $u := u(x,t)$ is the dual variable for (CE). To find the optimal solution μ^* for the minimum kinetic energy (18), we study the saddle point optimization problem

$$\min_{(\mu,\mathbf{v})\in\text{(CE)}} \max_u L(\mu,\mathbf{v},u), \tag{22}$$

where the minimization over $(\mu,\mathbf{v})$ runs over all flows satisfying (CE) such that $\mu(\cdot,0) = \mu_0$ and $\mu(\cdot,1) = \mu_1$. Note that if $\mu \notin$ (CE), then by scaling with arbitrarily large constant, we see that

$$\max_u \int_0^1 \int_{\mathbb{R}^d} (\partial_t \mu + \text{div}(\mu\mathbf{v}))\, u \, \mathrm{d}x \, \mathrm{d}t = +\infty. \tag{23}$$

Thus,

$$\min_{(\mu,\mathbf{v})\in\text{(CE)}} \int_0^1 \int_{\mathbb{R}^d} \frac{1}{2}||\mathbf{v}||_2^2 \mu \, dx \, dt = \min_{(\mu,\mathbf{v})} \max_u L(\mu,\mathbf{v},u) \tag{24}$$

$$\geqslant \max_u \min_{(\mu,\mathbf{v})} L(\mu,\mathbf{v},u), \tag{25}$$

where the minimization over $(\mu, \mathbf{v})$ is unconstrained. Using integration-by-parts and suitable decay for vanishing boundary as $\|x\|_2 \to \infty$, we have

$$L(\mu, \mathbf{v}, u) = \int_0^1 \int_{\mathbb{R}^d} \left[\frac{1}{2} \|\mathbf{v}\|_2^2 \mu - \mu \partial_t u - \langle \mathbf{v}, \nabla u \rangle \mu \right] \, \mathrm{d}x \, \mathrm{d}t \\ + \int_{\mathbb{R}^d} [\mu(\cdot, 1) u(\cdot, 1) - \mu(\cdot, 0) u(\cdot, 0)] \, \mathrm{d}x.$$

Now, we fix μ and u, and minimize $L(\mu, \mathbf{v}, u)$ over $\mathbf{v}$. The optimal velocity vector is $\mathbf{v}^* = \nabla u$, and we have

$$\max_u \min_\mu L(\mu, \mathbf{v}^*, u) = \int_0^1 \int_{\mathbb{R}^d} \left[- \left(\frac{1}{2} \|\nabla u\|_2^2 + \partial_t u \right) \mu \right] \, \mathrm{d}x \, \mathrm{d}t + \int_{\mathbb{R}^d} [u(\cdot, 1) \mu_1 - u(\cdot, 0) \mu_0] \, \mathrm{d}x, \tag{26}$$

for any flow μ_t satisfying the boundary conditions $\mu(\cdot, 0) = \mu_0$ and $\mu(\cdot, 1) = \mu_1$. If $\frac{1}{2}\|\nabla u\|_2^2 + \partial_t u \neq 0$, then by the same scaling argument above, we have

$$\min_\mu \int_0^1 \int_{\mathbb{R}^d} \left[- \left(\frac{1}{2} \|\nabla u\|_2^2 + \partial_t u \right) \mu \right] \, \mathrm{d}x \, \mathrm{d}t = -\infty \tag{27}$$

because μ is unconstrained (except for the boundary conditions). Then we deduce that

$$\min_{(\mu, \mathbf{v}) \in (\mathrm{CE})} \int_0^1 \int_{\mathbb{R}^d} \frac{1}{2} ||\mathbf{v}||_2^2 \mu \geqslant \max_{u \in (\mathrm{HJ})} \left\{ \int_{\mathbb{R}^d} u(\cdot, 1) \mu_1 - \int_{\mathbb{R}^d} u(\cdot, 0) \mu_0 \right\}, \tag{28}$$

where $u \in$ (HJ) means that u solves the *Hamilton-Jacobi equation* (HJ)

$$\partial_t u + \frac{1}{2} \|\nabla u\|_2^2 = 0. \tag{29}$$

From (28), we see that the duality gap is non-negative, and it is equal to zero if and only if (μ^*, u^*) solves the following system of PDEs

$$\begin{cases} \partial_t \mu + \operatorname{div}(\mu \nabla u) = 0, \;\; \partial_t u + \frac{1}{2} \|\nabla u\|_2^2 = 0, \\ \mu(\cdot, 0) = \mu_0, \;\; \mu(\cdot, 1) = \mu_1. \end{cases} \tag{30}$$

PDEs in (30) are referred to as the Karush–Kuhn–Tucker (KKT) conditions for the Wasserstein geodesic problem.

C Metric geometry structure of the Wasserstein space and geodesic

In this section, we review some basic facts on metric geometry properties of the Wasserstein space and geodesic. We first discuss the general metric space (X, d), and then specialize to the Wasserstein (metric) space $(\mathcal{P}_p(\mathbb{R}^d), W_p)$ for $p \geqslant 1$. Furthermore, we connect to the fluid dynamic formulation of optimal transport. Most of the materials are based on the reference books (Burage et al., 2001; Ambrosio et al., 2008; Santambrogio, 2015).

C.1 General metric space

Definition C.1 (Absolutely continuous curve). Let (X, d) be a metric space. A curve $\omega : [0, 1] \to X$ is *absolutely continuous* if there is a function $g \in L^1([0, 1])$ such that for all $t_0 < t_1$, we have

$$d(\omega(t_0), \omega(t_1)) \leqslant \int_{t_0}^{t_1} g(\tau) \, \mathrm{d}\tau. \tag{31}$$

Such curves are denoted by $\mathrm{AC}(X)$.

Definition C.2 (Metric derivative). If $\omega : [0, 1] \to X$ is a curve in a metric space (X, d), the *metric derivative* of ω at time t is defined as

$$|\omega'|(t) := \lim_{h \to 0} \frac{d(\omega(t + h), \omega(t))}{|h|}, \tag{32}$$

if the limit exists.

The following theorem generalizes the classical Rademacher theorem from a Euclidean space into any metric space in terms of the metric derivative.

Theorem C.3 (Rademacher). If $\omega : [0,1] \to X$ is Lipschitz continuous, then the metric derivative $|\omega'|(t)$ exists for almost every $t \in [0,1]$. In addition, for any $0 \leqslant t < s \leqslant 1$, we have

$$d(\omega(t), \omega(s)) \leqslant \int_t^s |\omega'|(\tau)\,\mathrm{d}\tau. \tag{33}$$

Theorem C.3 tells us that absolutely continuous curve ω has a metric derivative well-defined almost everywhere, and the "length" of the curve ω is bounded by the integral of the metric derivative. Thus, a natural definition of the length of a curve in a general metric space is to take the best approximation over all possible meshes.

Definition C.4 (Curve length). For a curve $\omega : [0,1] \to X$, we define its *length* as

$$\mathrm{Length}(\omega) := \sup\left\{\sum_{k=0}^{n-1} d(\omega(t_k), \omega(t_{k+1})) : n \geqslant 1, 0 = t_0 < t_1 < \ldots < t_n = 1\right\}. \tag{34}$$

Note that if $\omega \in \mathrm{AC}(X)$, then

$$d(\omega(t_k), \omega(t_{k+1})) \leqslant \int_{t_k}^{t_{k+1}} g(\tau)\,\mathrm{d}\tau \tag{35}$$

so that

$$\mathrm{Length}(\omega) \leqslant \int_0^1 g(\tau)\,\mathrm{d}\tau < \infty, \tag{36}$$

i.e., the curve ω is of bounded variation.

Lemma C.5. If $\omega \in \mathrm{AC}(X)$, then

$$\mathrm{Length}(\omega) = \int_0^1 |\omega'|(\tau)\,\mathrm{d}\tau. \tag{37}$$

Definition C.6 (Length space and geodesic space). Let $\omega : [0,1] \to X$ be a curve in (X, d).

1. The space (X, d) is a *length space* if
$$d(x, y) = \inf\left\{\mathrm{Length}(\omega) : \omega(0) = x, \omega(1) = y, \omega \in \mathrm{AC}(X)\right\}. \tag{38}$$

2. The space (X, d) is a *geodesic space* if
$$d(x, y) = \min\left\{\mathrm{Length}(\omega) : \omega(0) = x, \omega(1) = y, \omega \in \mathrm{AC}(X)\right\}. \tag{39}$$

Definition C.7 (Geodesic). Let (X, d) be a length space.

1. A curve $\omega : [0,1] \to X$ is said to be a *constant-speed geodesic* between $\omega(0)$ and $\omega(1)$ if
$$d(\omega(t), \omega(s)) = |t - s| \cdot d(\omega(0), \omega(1)), \tag{40}$$
for any $t, s \in [0,1]$.

2. If (X, d) is further a geodesic space, a curve $\omega : [0,1] \to X$ is said to be a *geodesic* between $x_0 \in X$ and $x_1 \in X$ if it minimizes the length among all possible curves such that $\omega(0) = x_0$ and $\omega(1) = x_1$.

Note that in a geodesic space (X, d), a constant-speed geodesic is indeed a geodesic. In addition, we have the following equivalent characterization of the geodesic in a geodesic space.

Lemma C.8. Let (X, d) be a geodesic space, $p > 1$, and $\omega : [0,1] \to X$ a curve connecting x_0 and x_1. Then the followings are equivalent.

1. ω is a constant-speed geodesic.

2. $\omega \in \mathrm{AC}(X)$ such that for almost every $t \in [0,1]$, we have
$$|\omega'|(t) = d(\omega(0), \omega(1)). \tag{41}$$

3. ω solves
$$\min\left\{\int_0^1 |\tilde{\omega}'|^p\,\mathrm{d}t : \tilde{\omega}(0) = x_0, \tilde{\omega}(1) = x_1\right\}. \tag{42}$$

C.2 Wasserstein space

Since the Wasserstein space $(\mathcal{P}_p(\mathbb{R}^d), W_p)$ for $p \geqslant 1$ is a metric space, the following definition specializes Definition C.2 to the Wasserstein metric derivative.

Definition C.9 (Wasserstein metric derivative). Let $\{\mu_t\}_{t\in[0,1]}$ be an absolutely continuous curve in the Wasserstein (metric) space $(\mathcal{P}_p(\mathbb{R}^d), W_p)$. Then the *metric derivative* at time t of the curve $t \mapsto \mu_t$ with respect to W_p is defined as

$$|\mu'|_p(t) := \lim_{h\to 0} \frac{W_p(\mu_{t+h}, \mu_t)}{|h|}. \tag{43}$$

For $p = 2$, we write $|\mu'|(t) := |\mu'|_2(t)$.

In the rest of this section, we consider probability measures μ_t that are absolutely continuous with respect to the Lebesgue measure on $\mathbb{R}^d$ and we use μ_t denote the probability measure, as well as its density, when the context is clear.

Theorem C.10. Let $p > 1$ and assume $\Omega \in \mathbb{R}^d$ is compact.

Part 1. If $\{\mu_t\}_{t\in[0,1]}$ is an absolutely continuous curve in $W_p(\Omega)$, then for almost every $t \in [0, 1]$, there is a velocity vector field $\mathbf{v}_t \in L^p(\mu_t)$ such that

1. μ_t is a weak solution of the continuity equation $\partial_t\mu_t + \mathrm{div}(\mu_t\mathbf{v}_t) = 0$ in the sense of distributions (cf. the definition in (49) below);
2. for almost every $t \in [0, 1]$, we have
$$\|\mathbf{v}_t\|_{L^p(\mu_t)} \leqslant |\mu'|_p(t), \tag{44}$$
where $\|\mathbf{v}_t\|^p_{L^p(\mu_t)} = \int_\Omega \|\mathbf{v}_t\|_2^p \,\mathrm{d}\mu_t$.

Part 2. Conversely, if $\{\mu_t\}_{t\in[0,1]}$ are probability measures in $\mathcal{P}_p(\Omega)$, and for each $t \in [0, 1]$ we suppose $\mathbf{v}_t \in L^p(\mu_t)$ and $\int_0^1 \|\mathbf{v}_t\|_{L^p(\mu)} \,\mathrm{d}t < \infty$ such that $(\mu_t, \mathbf{v}_t)$ solves the continuity equation, then we have

1. $\{\mu_t\}_{t\in[0,1]}$ is an absolutely continuous curve in $(\mathcal{P}_p(\mathbb{R}^d), W_p)$;
2. for almost every $t \in [0, 1]$,
$$|\mu'|_p(t) \leqslant \|\mathbf{v}_t\|_{L^p(\mu_t)}. \tag{45}$$

As an immediate corollary, we have the following dynamical representation of the Wasserstein metric derivative.

Corollary C.11. If $\{\mu_t\}_{t\in[0,1]}$ is an absolutely continuous curve in $(\mathcal{P}_p(\mathbb{R}^d), W_p)$, then the velocity vector field $\mathbf{v}_t$ given in Part 1 of Theorem C.10 must satisfy

$$\|\mathbf{v}_t\|_{L^p(\mu_t)} = |\mu'|_p(t). \tag{46}$$

Corollary C.11 suggests that $\mathbf{v}_t$ can be viewed as the *tangent vector field* of the curve $\{\mu_t\}_{t\in[0,1]}$ at time point t. Moreover, Corollary C.11 suggests the following (Euclidean) gradient flow for tracking particles in $\mathbb{R}^d$: let $y(t) := y_x(t)$ be the trajectory starting from $x \in \mathbb{R}^d$ (i.e., $y(0) = x$) that evolves according the ordinary differential equation (ODE)

$$\frac{\mathrm{d}}{\mathrm{d}t} y(t) = \mathbf{v}_t(y(t)). \tag{47}$$

The dynamical system (47) defines a flow $Y_t : \Omega \to \Omega$ of vector field $\mathbf{v}_t$ on Ω via

$$Y_t(x) = y(t). \tag{48}$$

Then, it is straightforward to check that the pushforward measure flow $\mu_t := (Y_t)_\sharp\mu_0$ and the chosen velocity vector field $\mathbf{v}_t$ in the ODE (47) is a weak solution of the continuity equation $\partial_t\mu_t + \mathrm{div}(\mu_t\mathbf{v}_t) = 0$ in the sense that

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_\Omega \phi \,\mathrm{d}t = \int_\Omega \langle \nabla\phi, \mathbf{v}_t\rangle \,\mathrm{d}\mu_t, \tag{49}$$

for any $\mathcal{C}^1$ function $\phi : \Omega \to \mathbb{R}$ with compact support.

Theorem C.12 (Constant-speed Wasserstein geodesic). Let $\Omega \in \mathbb{R}^d$ be a convex subset and $\mu, \nu \in \mathcal{P}_p(\Omega)$ for some $p > 1$. Let γ be an optimal transport plan under the cost function $\|x - y\|_p^p$. Define

$$\pi_t : \Omega \times \Omega \to \Omega,$$
$$\pi_t(x, y) = (1 - t)x + ty,$$

as the linear interpolation between x and y in Ω. Then, the curve $\mu_t = (\pi_t)_\sharp \gamma$ is a constant-speed geodesic in $(\mathcal{P}_p(\mathbb{R}^d), W_p)$ connecting $\mu_0 = \mu$ and $\mu_1 = \nu$.

If μ has a density with respect to the Lebesgue measure on $\mathbb{R}^d$, then there is an optimal transport map T from μ to ν (Brenier, 1991). According to Theorem C.12, we obtain *McCann's interpolation* (McCann, 1997) in the Wasserstein space as

$$\mu_t = [(1 - t)\text{id} + tT]_\sharp \mu, \tag{50}$$

which is a constant-speed geodesic in $(\mathcal{P}_p(\mathbb{R}^d), W_p)$. id is the identity function in $\mathbb{R}^d$.

To sum up, we collect the following facts about the geodesic structure and dynamical formulation of the OT problem. Let $p > 1$, and $\Omega \subset \mathbb{R}^d$ be a convex subset (either compact or have no mass escaping at infinity).

1. The metric space $(\mathcal{P}_p(\Omega), W_p)$ is a geodesic space.
2. For $\mu, \nu \in \mathcal{P}_p(\Omega)$, a constant-speed geodesic $\{\mu_t\}_{t\in[0,1]}$ in $(\mathcal{P}_p(\Omega), W_p)$ between μ and ν (i.e., $\mu_0 = \mu$ and $\mu_1 = \nu$) must satisfy $\mu_t \in \text{AC}(\mathcal{P}_p(\Omega))$ and
$$|\mu'|(t) = W_p(\mu(0), \mu(1)) = W_p(\mu, \nu) \tag{51}$$
for almost every $t \in [0, 1]$.
3. The above μ_t solves
$$\min \left\{ \int_0^1 |\tilde{\mu}'|^p(t) \,\mathrm{d}t : \tilde{\mu}(0) = \mu, \tilde{\mu}(1) = \nu, \tilde{\mu} \in \text{AC}(\mathcal{P}_p(\Omega)) \right\}. \tag{52}$$
4. The above μ_t solves the Benamou-Brenier problem
$$W_p^p(\mu, \nu) = \min \left\{ \int_0^1 \|\mathbf{v}_t\|^p_{L^p(\tilde{\mu}_t)} \,\mathrm{d}t : \tilde{\mu}(0) = \mu, \tilde{\mu}(1) = \nu, \partial_t \tilde{\mu}_t + \text{div}(\tilde{\mu}_t \mathbf{v}_t) = 0 \right\}, \tag{53}$$
and $\mu_t = \mu(\cdot, t)$ defines a constant-speed geodesic in $(\mathcal{P}_p(\Omega), W_p)$.

D Entropic regularization

Our GeONet is compatible with entropic regularization, which is closely related to the Schrödinger bridge problem and stochastic control (Chen et al., 2016). Specifically, the entropic-regularized GeONet (ER-GeONet) solves the following fluid dynamic problem:

$$\begin{gathered}\min_{(\mu,\mathbf{v})} \int_0^1 \int_{\mathbb{R}^d} \frac{1}{2} ||\mathbf{v}(x, t)||_2^2 \, \mu(x, t) \,\mathrm{d}x \,\mathrm{d}t \\ \text{subject to } \; \partial_t \mu + \text{div}(\mu \mathbf{v}) + \varepsilon \Delta \mu = 0, \;\; \mu(\cdot, 0) = \mu_0, \;\; \mu(\cdot, 1) = \mu_1.\end{gathered} \tag{54}$$

Applying the same variational analysis as in the unregularized case $\varepsilon = 0$ (cf. Appendix B), we obtain the KKT conditions for the optimization (54) as the solution to the following system of PDEs:

$$\partial_t \mu + \text{div}(\mu \nabla u) = -\,\varepsilon \Delta \mu, \tag{55}$$
$$\partial_t u + \frac{1}{2} \|\nabla u\|_2^2 = \varepsilon \Delta u, \tag{56}$$

with the boundary conditions $\mu(\cdot, 0) = \mu_0, \mu(\cdot, 1) = \mu_1$ for $\varepsilon > 0$. Note that (56) is a parabolic PDE, which has a unique smooth solution u^ε. The term $\varepsilon \Delta u$ effectively regularizes the (dual) Hamilton-Jacobi equation in (7). In the zero-noise limit as $\varepsilon \downarrow 0$, the solution of the optimal entropic interpolating flow (54) converges to solution of the Benamou-Brenier problem (4) in the sense of the method of vanishing viscosity (Mikami, 2004; Evans, 2010).

E GRADIENT ENHANCEMENT

In practice, we may fortify the base method by adding extra residual terms of the differentiated PDEs to our loss function of GeONet. Such gradient enhancement technique has been used to strengthen PINNs (Yu et al., 2022), which improves efficiency as fewer data points are needed to be sampled from $U(\Omega) \times U(0,1)$, and prediction accuracy as well.

The motivation behind gradient enhancement stems minimizing the residual of a differentiated PDE. We turn our attention to PDEs of the form

$$\begin{cases} \mathcal{F}\Big(x, t, \partial_{x_1} u, \dots, \partial_{x_d} u, \partial_{x_1 x_1} u, \dots, \partial_{x_d x_d} u, \dots, \partial_t u, \lambda\Big) = 0 & \text{on} \quad \Omega \times [0,1], \\ u(\cdot, 0) = u_0, \quad u(\cdot, 1) = u_1 & \text{on} \quad \Omega, \end{cases} \tag{57}$$

for domain $\Omega \subseteq \mathbb{R}^d$, parameter vector λ, and boundary conditions u_0, u_1. One may differentiate the PDE function $\mathcal{F}$ with respect to any spatial component to achieve

$$\frac{\partial}{\partial x_\ell}\mathcal{F}\Big(x, t, \partial_{x_1} u, \dots, \partial_{x_d} u, \partial_{x_1 x_1} u, \dots, \partial_{x_d x_d} u, \dots, \partial_t u, \lambda\Big) = 0. \tag{58}$$

The differentiated PDE is additionally equal to 0, similar to what we see in a PINN setup. If we substitute a neural network into the differentiated PDE of (58), what remains is a new residual, just as we saw with the neural network substituted into the original PDE. Minimizing this new residual in the related loss function characterizes the gradient enhancement method.

We utilize the same loss function in (14), but we add the additional terms

$$\mathcal{L}_{\text{GE,cty},i} = \frac{1}{N}\sum_{\ell=1}^{d} \gamma_\ell || \ \frac{\partial}{\partial x_\ell}(\frac{\partial}{\partial t}\mathcal{C}_{\phi,i} + \text{div}(\mathcal{C}_{\phi,i}\nabla\mathcal{H}_{\psi,i})) \ ||^2_{L^2(\Omega\times(0,1))}, \tag{59}$$

$$\mathcal{L}_{\text{GE, HJ},i} = \frac{1}{N}\sum_{\ell=1}^{d} \omega_\ell || \ \frac{\partial}{\partial x_\ell}(\frac{\partial}{\partial t}\mathcal{H}_{\psi,i} + \frac{1}{2}||\nabla\mathcal{H}_{\psi,i}||_2^2) \ ||^2_{L^2(\Omega\times(0,1))}, \tag{60}$$

where the variables and neural networks that also appeared in (14) are the same. Here γ_ℓ and ω_ℓ are positive weights. The summation is taken in order to account for gradient enhancement of each spatial component of $x \in \Omega$.

F DEEPONETS

A challenge resides in solving the risk minimization problem over numerous instances of data. This challenge may be conciliated by instituting a DeepONet that learns a general nonlinear operator, where one (or a pair of) neural network(s) encode(s) the input and another encodes the collocation samples. This architecture originates as an equivalence to the universal approximation theorem for operators.

General DeepONet. A general operator $G^\dagger$ may be approximated by an unstacked DeepONet (Chen and Chen, 1995; Lu et al., 2021)

$$G^\dagger(u_0)(x,t) \approx \sum_{k=1}^{p} \mathcal{B}_k\big(u_0(x_1), \dots, u_0(x_m), \theta\big) \cdot \mathcal{T}_k(x, t, \xi), \tag{61}$$

where $\mathcal{B}_k, \mathcal{T}_k$ are scalar elements of output of neural networks $\mathcal{B}, \mathcal{T}$, and p is a constant denoting the number of such elements. We take $\mathcal{B}$ and $\mathcal{T}$ to be artificial neural networks parameterized by θ, ξ respectively. $\mathcal{B}, \mathcal{T}$ are known as the branch and trunk networks respectively. u_0 is the initial function in which the operator is applied, evaluated at distinct locations $x_1, \dots, x_m$ for branch input. (x,t) is any arbitrary point in space and time in which $G^\dagger(u_0)$ may be evaluated.

Enhanced DeepONet. The above framework is restricted to one initial input function u_0. We turn our attention to the enhanced DeepONet, a DeepONet styled to act upon dual initial conditions Tan and Chen (2022). Our true operator $\Gamma^\dagger$ may be approximated using a second neural network encoder for input u_1,

$$\Gamma^\dagger(u_0, u_1)(x,t) \approx \sum_{k=1}^{p} \mathcal{B}_k^0\big(u_0(x_1), \ldots, u_0(x_m), \theta^0\big) \cdot \mathcal{B}_k^1\big(u_1(x_1), \ldots, u_1(x_m), \theta^1\big) \cdot \mathcal{T}_k(x,t,\xi). \tag{62}$$

Physics-informed DeepONet. The enhanced DeepONet may be substituted into any physics-informed framework, such as that of equation (9), taking place of the PDE solution value in the empirical loss to be minimized Wang et al. (2021a). Generalization of the trained DeepONet permits any solution to the PDEs to be evaluated instantaneously given the appropriate input function(s).

G Specialized aerchitectures

G.1 Modified multi-layer perceptron

Here we outline the forward pass of the modified multi-layer perceptron used throughout the bivariate and MNIST encoder experiments. Let σ denote an activation function (typically $\tanh$), X as neural network design input, W^i the weights of the neural network at index i, and b^i the bias at index i. Here, X can refer to either branch or trunk inputs, as this architecture is used for both.

The forward pass is given by

$$U = \sigma(W^1 X + b^1), \quad V = \sigma(W^2 X + b^2) \tag{63}$$
$$H^1 = \sigma(W^{h,1} X + b^{h,1}) \tag{64}$$
$$Z^k = \sigma(W^{z,k} H^k + b^{z,k}) \tag{65}$$
$$H^k = (1 - Z^{k-1}) \odot U + Z^{k-1} \odot U \tag{66}$$
$$\mathcal{N}_\theta = W^\ell H^\ell + b^\ell, \tag{67}$$

where $k \in \{1, \ldots, \ell\}$, $\odot$ is an element-wise product, and $\mathcal{N}_\theta$ is the neural network final output, either a branch or a trunk.

G.2 Fourier feature architecture

We outline the Fourier feature architecture of our empirical Gaussians experiment of 4.2. We leave the branches as multi-layer perceptrons, but we modify the trunks. In particular, we use

$$U_x = (\cos(2\pi B_x x), \sin(2\pi B_x x))^T \tag{68}$$
$$U_t = (\cos(2\pi B_t t), \sin(2\pi B_t t))^T \tag{69}$$
$$H_x^1 = \sigma(W_x^1 U_x + b_x^1) \tag{70}$$
$$H_t^1 = \sigma(W_t^1 U_t + b_t^1) \tag{71}$$
$$H_v^k = \sigma(W_v^k H_v^{k-1} + b_v^k), \quad v \in \{x, t\} \tag{72}$$
$$H^\ell = H_x^\ell \odot H_t^\ell \tag{73}$$
$$\mathcal{N}_\theta = W^\ell H^\ell + b^\ell, \tag{74}$$

where $k \in \{2, \ldots, \ell\}$, $\odot$ is an element-wise product, and $\mathcal{N}_\theta$ is the neural network trunk output. $B_v \in \mathbb{R}^{m \times d}$ are matrices, with the elements sampled from a $\mathcal{N}(0, \sigma_v^2)$ distribution. We denote d the dimension of space or time, and m is an arbitrary dimension inherent to the matrix, generally taken to be at least 100.

H Hyperparameter settings and training details

We discuss training characteristics of GeONet based on the primary experiments. An unmodified Adam optimizer was chosen for all branch, trunk neural networks with a learning rate starting from $5\text{e}{-4}$. All layers share the same width. We use $\tanh$ activation for all neural networks. Coefficients $\alpha_1, \alpha_2, \beta_0, \beta_1$ were computed after examining errors. Coefficients were selected in the range $[0.05, 20]$. Neural network depths refer to ℓ in each modified MLP. Training is done on a NVIDIA T4 GPU.

Table 5: Architecture and training details in our Gaussian mixture experiments of Section 4 and Appendix I.

Hyperparameter	1D Gaussians	2D Gaussians
No. of initial conditions (μ_0, μ_1)	20,000	5,000
m (branch input dimension)	100	576
Branch width	150	200
Branch depth	7	7
Trunk width	100	150
Trunk depth	7	7
p (dimension of outputs)	800	800
Batch size	2,000	2,000
Final training time	$\sim$ 2 hrs	$\sim$ 2 hrs
Final training loss	$\sim 1.5\text{e}{-4}$	$\sim 1.8\text{e}{-5}$
$\alpha_1, \alpha_2, \beta_0, \beta_1$	$0.5, 0.25, 1, 1$	$0.5, 0.25, 1, 1$

Table 6: Architecture and training details in our empirical Gaussians and encoded MNIST experiments of Section 4 and Appendix I.

Hyperparameter	Empirical Gaussians	Encoded MNIST
No. of initial conditions (μ_0, μ_1)	1,000	30,000
m (branch input dimension)	625	32
Branch width	100	150
Branch depth	7	7
Trunk width	100	100
Trunk depth	5	7
p (dimension of outputs)	200	200
Batch size	1,000	1,000
Final training time	$\sim$ 2 hrs	$\sim$ 4 hrs
Final training loss	$\sim 7.0\text{e}{-4}$	$\sim 2.0\text{e}{-2}$
$\alpha_1, \alpha_2, \beta_0, \beta_1$	$0.5, 0.25, 1, 1$	$1, 1, 1, 1$

I Training and performance

I.1 Univariate and bivariate Gaussian mixture experiments

Univariate Gaussians. We choose spatial domain $x \in \Omega = [0, 10]$ discretized into a 100 point mesh. We generate $20,000$ training pairs (μ_0, μ_1) of Gaussians, taking $k_j = 6$ for the number of Gaussians in each mixture. We take means $\mu_i \in [2, 8]$ and variances $\Sigma_i \in [0.5, 0.6]$ uniformly. Empirically, we found a large batch size more suitable for training than a low one, and so we take a batch size of $2,000$, meaning this many uniform collocation points are taken for both the PDE residuals and boundary points for each training iteration. We choose physical loss coefficient $\alpha_1 = 0.5, \alpha_2 = 0.25$, with boundary coefficients $\beta_0 = \beta_1 = 1$. We found these coefficients a good balance to enforce the physical constraint without sacrificing boundary restrictions after iterating these coefficients among $[0.05, 20]$ and examining error. Additional training details are given in Appendix H.

Bivariate Gaussians. In our experiment, domain $\Omega = [0, 5] \times [0, 5] \subseteq \mathbb{R}^2$ was chosen, which was discretized into a 24×24 grid for GeONet input, meaning the branch networks took vector input of 576 in length for each. We generate $5,000$ training pairs (μ_0, μ_1). Recall that GeONet is mesh-invariant, so the 24×24 grids can be adapted to any higher resolution, which is used in figure 7. We use a combination of low and high variance Gaussians in the mixture, 6 of which had variance in $[0.35, 0.4]$ and 6 in $[0.75, 0.9]$, giving a total of 12 Gaussians in each mixture in each pair. Covariance was in $[-0.1, 0.1]$. Additional training details are given in Appendix H.

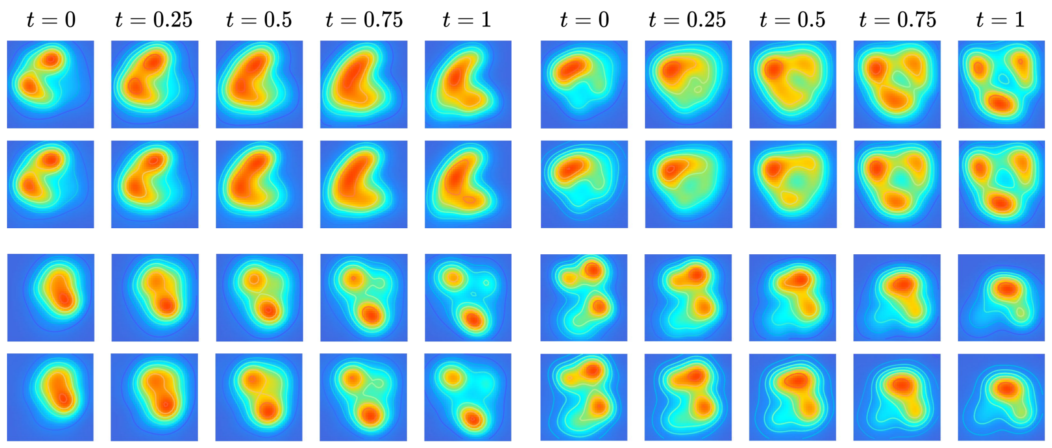


Figure 7: Geodesics predicted by GeONet on bivariate Gaussians over a square domain. The top of each pair is the reference solution computed by POT, and the bottom is GeONet.

Training. To compute the DeepONet derivatives, we take the inner product in the enhanced DeepOnet as in equations (12), (13), and subsequently use automatic differentiation after the inner products are taken. Alternatively, one may compute a Hessian for the second-order derivatives, but this is costly in terms of memory, meaning a large batch size cannot be used without a monumental memory cost. We found the neural networks do not train properly without a large batch size, and so this method of differentiation is not viable. We found the DeepONet output dimension taken to be quite large slightly outperforms a lower-dimension output given sufficient data and no overfitting. In the univariate Gaussian experiment, we take $p = 800$, which outperformed $p = 200$ by reducing training loss from approximately $2.5\text{e}{-4}$ to $1.5\text{e}{-4}$ and reducing test error by about 1%. In the bivariate experiment, changing $p = 400$ to $p = 800$ reduced training loss from approximately $2.1\text{e}{-5}$ to $1.8\text{e}{-5}$. Architecture generally made some difference to training loss, but not significant, making a width of around 100-200 suitable for branches and trunks. For example, increasing branch width in the univariate experiment from 100 to 150 lowered training loss by approximately $4\text{e}{-5}$. Increasing branch width to 200 and trunk width to 150 from 150 and 100 respectively had minimal effect, lowering training loss by about $1\text{e}{-5}$. We found the modified MLP architecture preferable, lowering final training loss from approximately $3\text{e}{-4}$ with standard architecture for univariate Gaussians.

Performance. Our baseline results were collected by deploying GeONet on the identity geodesic in Table 2. The baseline identity geodesic provides a benchmark for comparing and interpreting the errors across different setups. The univariate cases were evaluated upon a 100 point mesh, and the bivariate upon a 40×40 mesh, except in the zero-shot super resolution case, in which the grid is refined and previously specified. From Table 2, we can draw the following observations. The loss boundary conditions (17) allow greater precision for $t = 0, 1$, which suggests that lack of data-enforced conditions along the inner region of the time continuum would cause greater error. Errors for predicting the univariate Gaussian trivial identity geodesic in the intermediate $t = 0.25, 0.5, 0.75$ are uniformly smaller than other in-distribution setups since the former is an easier task. In the bivariate experiment, we found that error quickly rises as variance decreases, which is equivalent to a task of learning more complicated geodesics. We did not find lower variance drastically affects performance in the univariate experiment, suggesting GeONet and potentially physics-informed DeepONets in general are less effective as dimension increases. We did not find the number of Gaussians in the mixtures drastically affected results, but naturally more complicated geodesics induce greater error, which is to be expected. We found bivariate errors are similar to the random case as in the identity case, suggesting there is some notion of base neural operator error, which may not exist with simpler data.

I.2 Gaussian empirical densities

Training. 3000 point cloud particles were sampled from mixtures composed of 3 Gaussians for source μ_0 and target μ_1. 2D histograms were constructed to turn particle data into empirical densities, with bins ranging from -7 to 7. Domain $\Omega = [0, 5] \times [0, 5]$ was discretized into a 25×25 point domain and assigned for the histograms' locations used as GeONet spatial input. A batch size of 1,000 was chosen. We take $p = 200$, $\alpha_1 = 0.5, \alpha_2 = 0.25, \beta_0 = \beta_1 = 1$, which can be altered to impose strength of the boundary and physics terms accordingly. We employ the Fourier feature network architecture of Wang et al. (2021b) for trunk networks. We take matrix B_v with elements sampled in $\mathcal{N}(0, \sigma_v^2)$, subsequently taking $(\cos(2\pi B_v v), \sin(2\pi B_v v))^T$ as input for a fully-connected network, where v is either space or time input. Our architecture for this experiment is fully outlined in G.2. Empirically, we found low variance necessary, and we chose $\sigma = 0.5$ for both $v = x, t$ for both continuity and trunk branches.

Performance. In this experiment, GeONet is the only method among the comparison which correctly captures the geodesic behavior. While the other methods are suitable for point clouds specifically, they fail to capture the geodesic. This is demonstrated by our errors in table 3, as the errors of GeONet are significantly smaller than those of other methods. GeONet tends to slightly regularize the solutions, but the Fourier feature embedding does help correct this by being able to introduce more detail, such as the randomness from the samples, into the solution, such as with the initial conditions. Generally, we found higher variance of B_v failed, as the physics-informed terms fail. For example, the Hamilton-Jacobi equation learns the trivial solution. High variance generally is able to capture detail to a better degree than low variance in the Fourier feature architecture Tancik et al. (2020). A possible remedy for this is to train the networks on low variance, then fix the Hamilton-Jacobi equation once converged. Then, one may possibly retrain only the continuity networks.

I.3 MNIST experiment

Training. To learn the geodesic, we ensure all values within the encoded representation are nonnegative, meaning we can shift all encoded representations by some arbitrary constant. We choose 10 for this. This constant can be deducted in later stages to ensure the valid represenation is met. A domain of $[0, 5]$ was dividied into an equispaced mesh of 32 points for the encoded representation. This domain is rather arbitrary and is chosen simply for DeepONet input purposes, which can be modified as seen fit. $5,000$ encoded pairs were chosen to train GeONet, but the pretrained autoencoder was done on the entirety of MNIST with a batch size of 500 and 120 iterations per epoch. Additional training details are found in Appendix H.

Performance. GeONet performs well in this experiment. Scaling the physics-informed term by a constant less than one did not prove necessary in this experiment to ensure all loss terms are met to a sufficient degree. As before, boundary terms are uniformly smaller, likely since these terms are known and included in the loss function to be minimized. The same error metric is used as in the synthetic

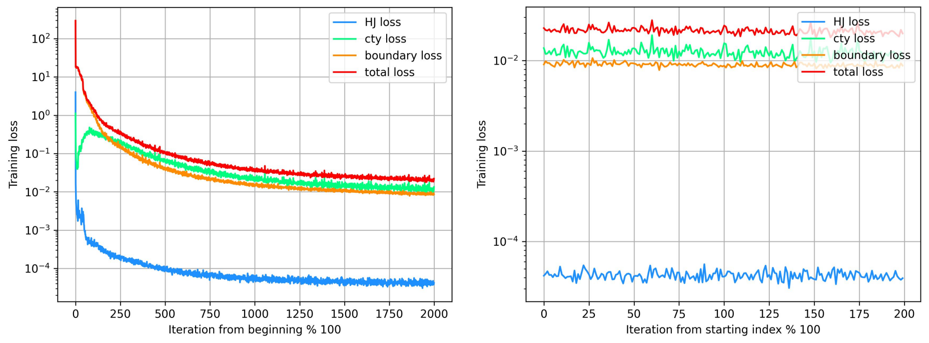


Figure 8: We examine iterations of the Adam optimizer in the total and late training on a log scale. The learning rate was modified every iteration. We examine late training in order to observe oscillatory behavior between the continuity and Hamilton-Jacobi loss to see if they adversarially compete in late training. We do not observe this pattern, and the continuity loss greatly surpasses the HJ loss in value. These graphs were created using the encoded MNIST experiment.

experiments but with normalization, making the L^1 error relative. We remark OOD generalization is omitted because the distribution of the encoded data is not known. We also remark the decoded images, being the geodesic returned to its original state, do not directly translate to a geodesic performed upon an original pair of images. NaN values are omitted in the error computations, which are possible in the POT solutions due to the irregularity of the initial conditions.

Regularization. Classical geodesic algorithms require a small regularization parameter in order to be computed. This provides affects the synthetic experiments trivially, but we found this regularization induces greater in the MNIST experiment. This is to be considered when evaluating the errors, and true error is likely smaller between GeONet and the reference geodesics computed with POT than what is displayed. This regularization acts as a form of "smoothing" of the solutions.

I.4 Out-of-distribution generalization.

To examine the generalization performance of GeONet under distribution shifts, we consider a setup which is out-of-distribution (OOD) at test time. To do this, we alter the variances to be lower than in the training setup. Changing the number of Gaussians in the mixture at test time does not yield a significant enough of an alteration to the experiment. For univariate Gaussians, variance was among $[0.3, 0.4]$ instead of the $[0.5, 0.6]$ in training. For bivariate Gaussians, we take variances in $[0.25, 0.3]$ and $[0.65, 0.8]$, six of each in each mixture. Empirically, we found OOD generalization had lower error in this experiment than in the univariate case, which may be because the support of areas of low variance have relatively low measure. This bivariate error does indeed scale greater as variance decreases even more. We remark the lower variance generally yields a more sophisticated task in learning the geodesic, and error even trained upon such data is higher than with high variance.

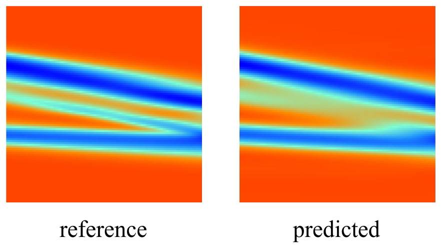


Figure 9: An illustration of an OOD univariate Gaussian mixture geodesic at test time.

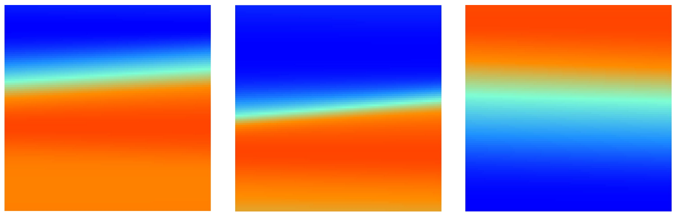
J GeONet error for additional error metrics

Error metric. We use the L^1 error $\int_\Omega |\mathcal{C} - \mu| \mathrm{d}x$ as our error metric to assess the performance, where $\mu := \mu(x, t)$ is a reference geodesic as proxy of the true geodesic without entropic regularization. The L^1 error integral is estimated by evaluating a discrete Riemann sum along a mesh and the reference is computed using the Convolutional Wasserstein Barycenter framework within the POT Python library (Solomon et al., 2015; Flamary et al., 2021). Since $\int_\Omega |\mu| \mathrm{d}x = 1$ for all time points, the L^1 error is also a relative error, a meaningful metric essentially corresponding to the percentage error between the neural operator geodesic and the reference. We also consider the L^2 and Wasserstein error metric for predicted Wasserstein geodesics.

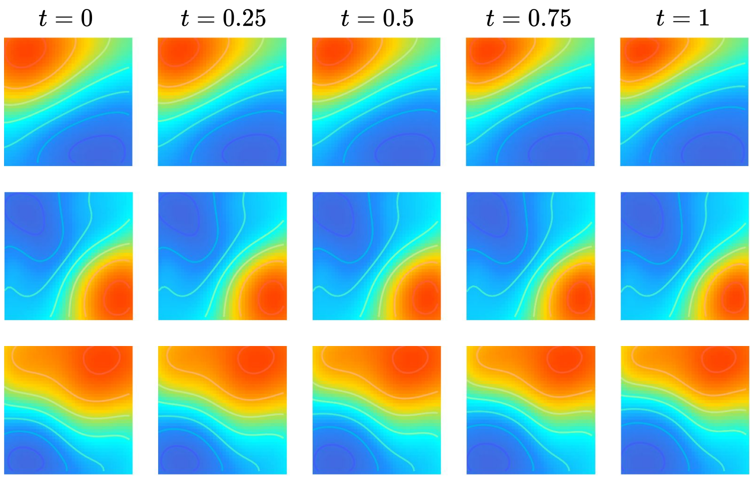
Table 7: We list mean and standard deviations of error of GeONet on 50 random $\mu_0 \neq \mu_1$ samples for alternative error metrics, being L^2 error and the Wasserstein-1 distance. We remark we use sliced Wasserstein distance for the 2D case, as this metric is computationally feasible for higher dimensional cases. We perform this for random Gaussian mixture pairings. All values are multiplied by 10^{-2} by those of the table.

	GeONet alternative metric error for random Gaussian mixtures				
Experiment	$t = 0$	$t = 0.25$	$t = 0.5$	$t = 0.75$	$t = 1$
1D, L^2	5.19 ± 1.74	6.91 ± 4.81	7.28 ± 5.39	6.49 ± 4.36	4.81 ± 1.58
1D, Wasserstein	0.352 ± 0.116	0.364 ± 0.178	0.403 ± 0.228	0.386 ± 0.166	0.347 ± 0.101
2D, L^2	6.93 ± 0.883	7.72 ± 1.23	8.11 ± 1.30	7.79 ± 1.14	6.87 ± 1.05
2D, Wasserstein	0.245 ± 0.0329	0.264 ± 0.0316	0.275 ± 0.0447	0.267 ± 0.0338	0.246 ± 0.0356

K SAMPLE HJ GRAPHS



(a)



(b)

Figure 10: We present sample HJ equations for (a) three univariate Gaussian mixtures and (b) three bivariate Gaussian mixtures from the primary experiments performed in Section 4. The univariate HJ samples at certain times are the vertical cross sections of the graphs, and the bivariate samples are given at certain times.