A. Brief Discussion of Convergence and Guarantees

Proving existence and uniqueness of solutions of gradient flows in Wasserstein space, or convergence of their discretized schemes, is challenging. But it can be achieved through different types of assumptions on the spaces, metrics, and functionals. Here, we will briefly discuss guarantees that depend on one of the simplest such assumptions: geodesic convexity.

Definition A.1. *Let $\mathcal{X}$ be a geodesic metric space and $F : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ a functional. We say that F is (λ-)**geodesically convex** if it is (λ-)convex along geodesics in $\mathcal{X}$, i.e., for every pair of points $x_0, x_1 \in \mathcal{X}$, there exists a constant-speed geodesic ω connecting $\omega(0) = x_0$ to $\omega(1) = x_1$ such that $t \mapsto F(\omega(t)), t \in [0, 1]$ is (λ-)convex.*

Note that if $\mathcal{X}$ is Euclidean, the definition above is simply λ-convexity. On the other hand, this concept is well defined for metric measure spaces like $\mathbb{W}_p(\Omega)$ too. In particular, for $\mathbb{W}_2(\Omega)$, all geodesics are displacement geodesics, so the condition above is also known as *displacement convexity*.

Thus, a functional $F : \mathbb{W}_p(\mathcal{X}) \to \mathbb{R} \cup \{+\infty\}$ is λ-geodesically convex if and only if for every pair $\mu^1, \mu^2 \in \mathcal{P}(\mathcal{X})$ there exists an optimal transport coupling $\pi \in \Pi(\mu^1, \mu^2)$ such that

$$F(\mu_t^{1\to 2}) \leq (1-t)F(\mu^1) + tF(\mu^2) - \frac{\lambda}{2}t(1-t)\mathrm{W}_p^2(\mu^1, \mu^2) \quad \forall t \in [0, 1] \tag{18}$$

where $\mu_t^{1\to 2} \triangleq ((1-t)x + ty)_\sharp \pi$ is a geodesic in $\mathbb{W}_p(\mathcal{X})$ interpolating between μ^1 and μ^2.

It can be shown that the functionals used in this work (i.e., those in Eq. (9)) are displacement convex under suitable conditions (Santambrogio, 2017). Specifically, $\mathcal{V}$ and $\mathcal{W}$ are λ-displacement convex if the underlying potentials V and W are λ-convex. For the internal energy functional $\mathcal{F}$, some technical assumptions on f are needed, such as requiring that $f(0) = 0$, $s \mapsto s^d f(s^{-d})$ is convex and decreasing, and the underlying space is convex (Santambrogio, 2017, Thm 7.28). It is easy to see that simple functions, such as the entropy term discussed before, or power functions with exponent $q > 1$, all satisfy this condition.

The following result, one of the simplest in such family of guarantees, shows the crucial importance of λ-geodesic convexity for establishing guarantees of gradient flows in Wasserstein space:

Proposition A.2 (Santambrogio 2017, Prop 4.13)**.** *Suppose that $F : \mathbb{W}_2(\mathcal{X}) \to \mathbb{R} \cup \{+\infty\}$ is λ-geodesically convex and that the two curves ρ_t^0 and ρ_t^1 are solutions of (8). Then, setting $\delta(t) = \frac{1}{2}\mathrm{W}_2^2(\rho_t^1, \rho_t^2)$, we have*

$$\delta'(t) \leq -2\lambda d(t)$$

This implies uniqueness of the solution of (8) for a fixed initial state, stability and exponential convergence of the flow as $t \to +\infty$ if $\lambda > 0$.

Unfortunately —and somewhat unexpectedly— the functional $\mathcal{T}_\beta(\rho) = \mathrm{W}_2^2(\rho, \beta)$ turns out to be not displacement convex in general. However, it does satisfy an alternate and more general notion of convexity: along generalized geodesics.

Definition A.3. *Let $\rho \in \mathcal{P}(\mathcal{X})$ be fixed. For every pair $\mu^1, \mu^2 \in \mathcal{P}(\mathcal{X})$, a **generalized geodesic** between them with base ρ in $\mathbb{W}_2(\mathcal{X})$ is given by the curve $\mu_t = \big((1-t)T_0 + tT_1\big)_\sharp \rho$ where T_i is the optimal transport map (for the squared cost) from ρ to μ^i.*

Thus, a functional $F : \mathbb{W}_p(\mathcal{X}) \to \mathbb{R} \cup \{+\infty\}$ is λ-geodesically convex along generalized geodesics if it satisfies condition (18) for *generalized* geodesics. Under the same assumptions as above, the functionals $\mathcal{V}, \mathcal{W}$, and $\mathcal{F}$ are all convex along generalized geodesics too (Santambrogio, 2017; Ambrosio et al., 2005). But now, as hinted at before, so is $\mathcal{T}_\beta(\rho)$ if we choose β as the base point of the generalized geodesics (Santambrogio, 2015).

The notion of convexity along generalized geodesics can be used to establish results analogous to Proposition A.2 but which apply to more general functionals, including $\mathcal{T}_\beta(\rho)$. Such results usually involve appealing to a characterization of gradient flows known as the evolution variational inequality (EVI):

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{1}{2}d(\rho_t, \beta) \leq F(\beta) - F(\rho_t) - \frac{\lambda}{2}d(\mu_t, \beta)^2 \qquad \forall \beta \in \mathcal{P}(\mathcal{X}) \tag{19}$$

Convexity along generalized geodesics can be used to prove the EVI conditions holds for a certain functional, which in turn implies uniqueness and stability of the flow. We refer the reader to (Santambrogio, 2017) for further details.

B. First Variations, Gradient Flows, and Connections to PDEs.

B.1. First variation of a functional

As mentioned in Section 4.1, having a notion of derivative of functionals over measures is a crucial step towards defining gradient flows in that space. The notion we rely on here is that of first variation of a functional (Santambrogio, 2017):

Definition B.1. *Given a functional $F : \mathcal{P}(\Omega) \to \mathbb{R}$, consider perturbations χ such that at least for every $\epsilon \in [0, \epsilon_0]$, $\rho + \epsilon\chi \in \mathcal{P}(\Omega)$. If there exists a function G such that*

$$\frac{\mathrm{d}}{\mathrm{d}\epsilon} F(\rho + \epsilon\chi)\Big|_{\epsilon=0} = \int G(\rho)\,\mathrm{d}\chi$$

*for every such perturbation χ, we call it the **first variation** of F at ρ, and denote it by $\frac{\delta F}{\delta \rho}$.*

B.2. Gradient flows and PDEs

The connection between OT and certain diffusive partial differential equations (PDE) has been well studied over the past two decades (Jordan et al., 1998; Otto, 2001). Indeed, equation (8) defines a PDE over densities ρ. As mentioned before, it has a fluid dynamics interpretation as a continuity equation on a density-dependent flow velocity vector field $\mathbf{u} \triangleq -\nabla \frac{\delta F}{\delta \rho}(\rho)$, or a conservation-of-energy PDE for the energy flux $\mathbf{q} \triangleq -\rho \nabla \frac{\delta F}{\delta \rho}(\rho)$. In the context of densities and datasets, this PDE can be roughly understood as a conservation-of-mass principle: no probability mass is created or destroyed in the sequence of densities on $\mathcal{X} \times \mathcal{Y}$ that solve this system.

For a functional of the form (12) with only $\mathcal{F}, \mathcal{V}, \mathcal{W}$ terms, the corresponding PDE (eq. (14)) is known as a diffusion–advection–interaction equation. Certain choices of functionals $\mathcal{F}, \mathcal{V}, \mathcal{W}$ recover familiar PDEs. For example, taking $F(\rho) = \mathcal{F}(\rho) + \mathcal{V}(\rho)$, and $f(t) = t \log t$, the gradient flow of F solves a Fokker-Planck equation (Santambrogio, 2015):

$$\partial_t \rho - \Delta\rho - \nabla \cdot (\rho \nabla V) = 0.$$

In dataset space, this equation can be interpreted as the time evolution of a dataset subject to a drift force imposed by the potential function V and a constant-variance diffusion term ($\Delta\rho$) resulting from the entropy-inducing functional $\mathcal{F}$. Other choices of functionals allow us to recover the advection equation, porous-media equation, and various other diffusion–advection–interaction PDEs (Santambrogio, 2017). As we did for the Fokker-Planck equation, interpreting these PDEs in our context of dataset dynamics might yield interesting insights for designing objective functions.

C. Implementation and Experimental Details

We implement our method on `PyTorch` (Paszke et al., 2019), using the `geomloss` (Feydy et al., 2019) and `POT` (Flamary et al., 2021) libraries for OT-related computations, including the OTDD distance needed at every step. The three types of feature-label dynamics described in Section 6 are implemented by detaching parts of the computational graph in order to make gradient updates only in some of them. For the variable label dynamics, there are two options for clustering: fixed-size or nonparametric. We use k-means for the former and density-based spatial clustering of applications with noise (DBSCAN) with parameters $\epsilon = 5$ and minimum points per cluster 4 for the latter. Pseudocode for the three types of feature-label gradient flow dynamics described in Section 6 is shown here in Algorithms 1 to 3.

For the parametrized flow mapping h_{flow} (§7.2), we use an autoencoder-type architecture with an encoder consisting of 2 convolutional and 5 fully-connected layers, and the decoder is a inverted copy of the encoder. It was trained for 20 epochs using ADAM with learning rate 1×10^{-3}, using ten different random restarts and choosing the best performing one in a held-out set. For transfer learning (§7.2), we use a LeNet-5 architecture with ReLU nonlinearities trained for 20 epochs using ADAM with learning rate 1×10^{-3} and weight decay 1×10^{-6} It was fine-tuned for 10 epochs on the target domain(s) using the same optimization parameters. For the experiments in Table 1, we use 5K source (MNIST) and target (other *NIST datasets) samples. For both supervised and unsupervised flows, we use a flow step size of 1×10^{-1}, 1000 steps, and entropy regularization $\lambda = 1 \times 10^{2}$ For the unsupervised flow, we permute the values of the pseudo-labels obtained through clustering to match them to the indices of the target labels so as to allow accuracy computation.

All experiments were run on the same machine with an Intel Xeon 32-core 2.00GHz CPU with a single GeForce RTX 2080 Ti GPU. In this machine, the flows on synthetic datasets of Section 6 run at <0.2s per step, while the flows for the image classification datasets of Sections 6 and 7.2 run at $\sim$ 5s per step for 2K particles, for a total flow runtime of less than 5 minutes. Information about all the datasets used, including references, are provided in Table 2.

Algorithm 1 Gradient flow with feature-driven fixed-label (`fd-fl`) dynamics.

Input: Initial particle feature matrix $X_0 \in \mathbb{R}^{d \times n}$ and corresponding labels $\mathbf{y} \in \{0, \dots, k\}^n$.
requires_gradient(X_0) $\leftarrow$ `True`
for time $t = 0$ **to** T **do**
  $\ell \leftarrow F(\mathbf{X}_t, \mathbf{y})$
  $\mathbf{X}_t \leftarrow$ optim_step($\nabla_{\mathbf{X}}\, \ell$)
  for every class $j = 1$ **to** k **do**
    $\mu_t^j, \Sigma_t^j \leftarrow$ getstats($\{\mathbf{x}_t^i \mid y_i = j\}$)
  end for
  recompute_label_distances($\{\mu_t^j\}, \{\Sigma_t^j\}$) {subroutine in OTDD §3.3}
end for

Algorithm 2 Gradient flow with joint-driven fixed-label (`jd-fl`) dynamics.

Input: Initial particle feature matrix $X_0 \in \mathbb{R}^{d \times n}$ and corresponding labels $\mathbf{y} \in \{0, \dots, k\}^n$.
requires_gradient(X_0, Σ_0^j, μ_0^j) $\leftarrow$ `True`
for time $t = 0$ **to** T **do**
  $\ell \leftarrow F(\mathbf{X}_t, \mathbf{y})$
  $\mathbf{X}_t \leftarrow$ optim_step($\nabla_{\mathbf{X}}\, \ell$)
  for every class $j = 1$ **to** k **do**
    $\mu_t^j \leftarrow$ optim_step($\nabla_{\mu^j}\, \ell$)
    $\Sigma_t^j \leftarrow$ optim_step($\nabla_{\Sigma^j}\, \ell$)
  end for
  recompute_label_distances($\{\mu_t^j\}, \{\Sigma_t^j\}$) {subroutine in OTDD §3.3}
end for

Algorithm 3 Gradient flow with joint-driven variable-label (`jd-vl`) dynamics.

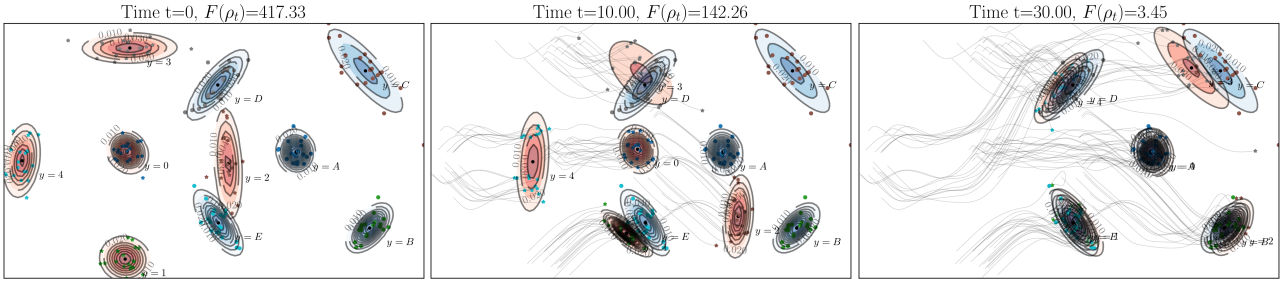
Input: Initial particle feature matrix $X_0 \in \mathbb{R}^{d \times n}$ and corresponding labels $\mathbf{y} \in \{0, \dots, k\}^n$.
requires_gradient(X_0, Σ_0^j, μ_0^j) $\leftarrow$ `True`
for time $t = 0$ **to** T **do**
  $\ell \leftarrow F(\mathbf{X}_t, \mathbf{y})$
  $\mathbf{X}_t \leftarrow$ optim_step($\nabla_{\mathbf{X}}\, \ell$)
  for every particle $i = 1$ **to** n **do**
    $\mu_t^i \leftarrow$ optim_step($\nabla_{\mu^i}\, \ell$)
    $\Sigma_t^i \leftarrow$ optim_step($\nabla_{\Sigma^i}\, \ell$)
  end for
  $\mathbf{y}_t \leftarrow$ clustering_method($\{\mu_t\}, \{\Sigma_t\}$) {recompute discrete labels by clustering}
  recompute_label_distances($\{\mu_t^j\}, \{\Sigma_t^j\}$) {subroutine in OTDD §3.3}
end for

Table 2. Summary of datasets used. $*$: we rescale the USPS digits to 28×28 for comparison to the *NIST datasets, and the STL-10 and CAMELYON to 32×32 for comparison to CIFAR-10.

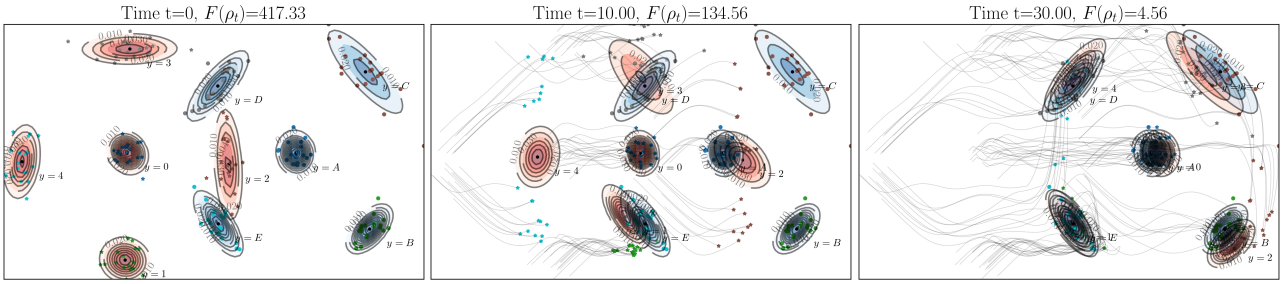
Dataset	Input Dimension	Number of Classes	Train Examples	Test Examples	Source
USPS	$16 \times 16^*$	10	7291	2007	(Hull, 1994)
MNIST	28×28	10	60K	10K	(LeCun et al., 2010)
KMNIST	28×28	10	60K	10K	(Clanuwat et al., 2018)
FASHION-MNIST	28×28	10	60K	10K	(Xiao et al., 2017)
CAMELYON	$128 \times 128^*$	2	262K	32K	(Litjens et al., 2018)
CIFAR-10	32×32	10	50K	10K	(Krizhevsky & Hinton, 2009)
STL-10	$96 \times 96^*$	10	5K	8K	(Coates et al., 2011)

D. Additional Experimental Results on Gaussian Flows

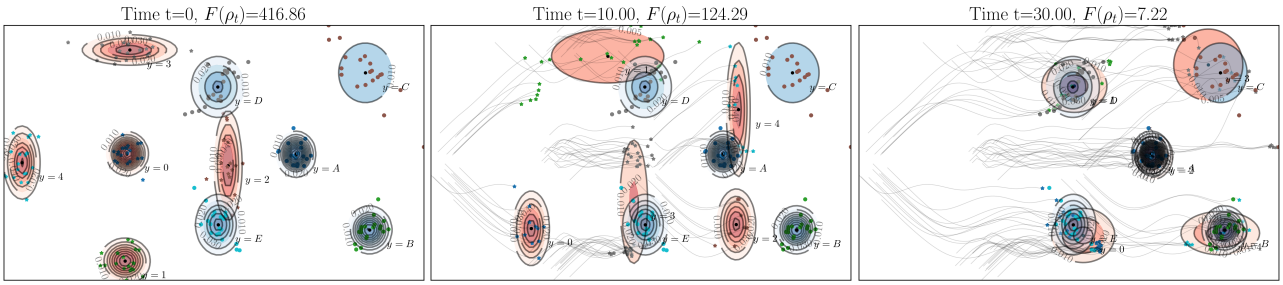
For the simple synthetic dataset example of Section 6, we show a comparison of the three types of flow dynamics in Figure 7, and experiments with various types of functionals in Figure 8.



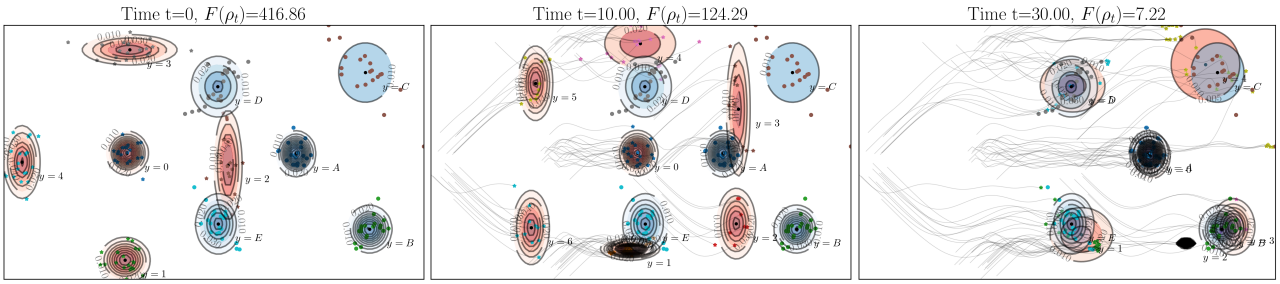
(a) Feature-driven (`fd`) dynamics, ADAM optimizer.



(b) Joint-driven fixed-label (`jd-fl`) dynamics, ADAM optimizer.

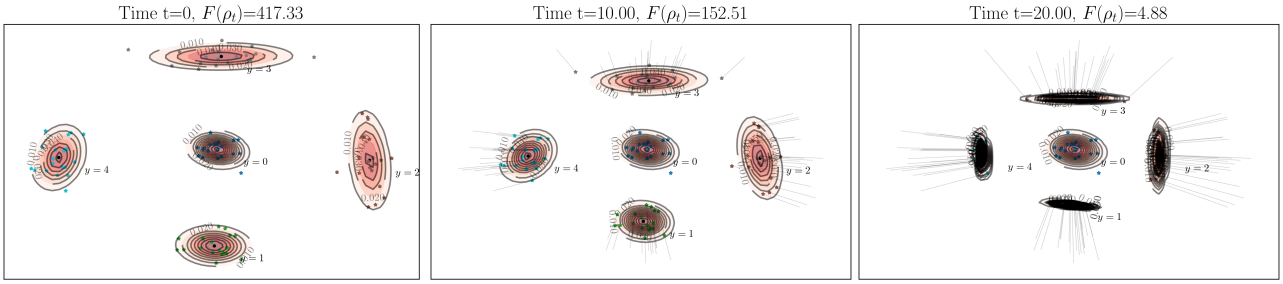


(c) Joint-driven variable-label (`jd-vl`) dynamics, k-means clustering, ADAM optimizer.

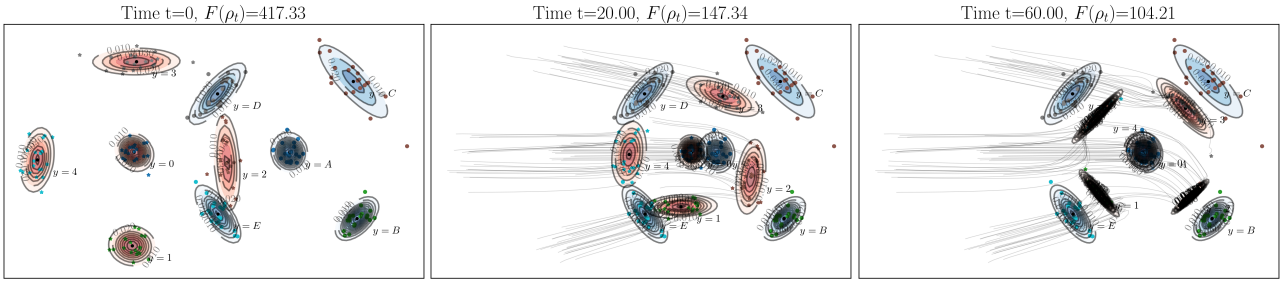


(d) Joint-driven variable-label (`jd-vl`) dynamics, DBSCAN clustering, ADAM optimizer.

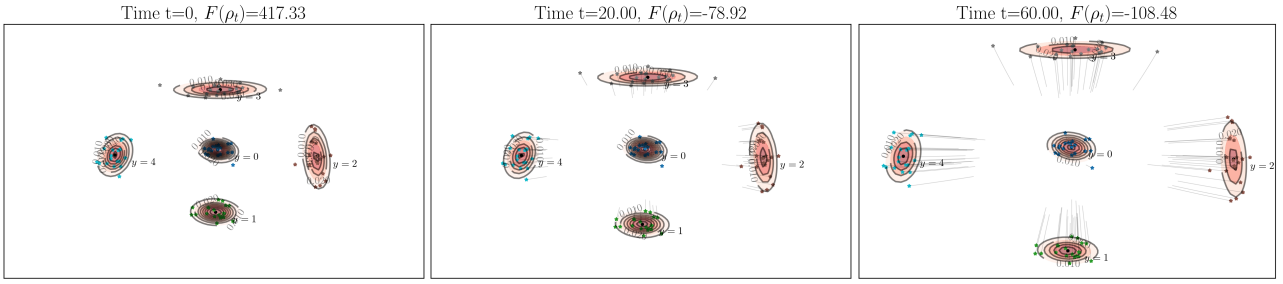
Figure 7. Gradient flows driven by functional $\mathcal{T}_\beta(\rho) = \text{OTDD}(\text{D}_\rho, \text{D}_\beta)$ starting from dataset ρ_0 (red) advecting towards β (blue) for various dynamic schemes (§6).



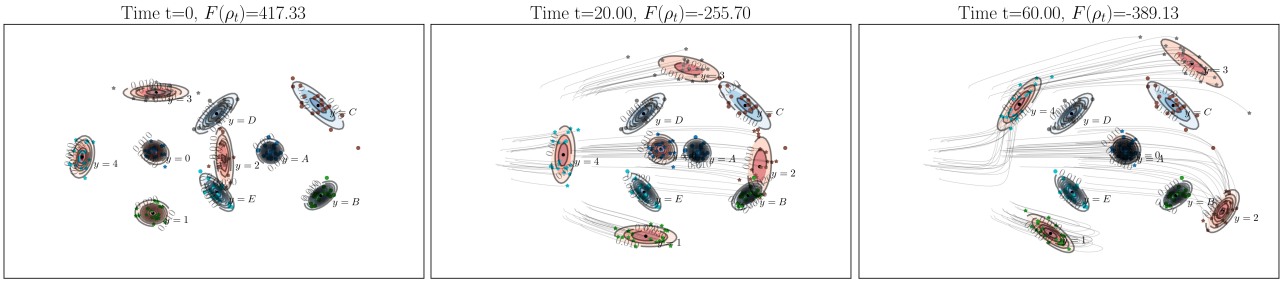
(a) Functional: $F(\rho) = \int (\|\mathbf{x} - \mathbf{x}_0\| - \tau)_+ \, \mathrm{d}\rho(z)$, SGD optimizer.



(b) Functional: $F(\rho) = \mathrm{OTDD}(\mathrm{D}_\rho, \mathrm{D}_\beta) + \lambda \int (\|\mathbf{x} - \mathbf{x}_0\| - \tau)_+ \, \mathrm{d}\rho(z)$, SGD optimizer.



(c) Functional: $F(\rho) = \iint -\|\mathbf{x} - \mathbf{x}'\|^2 \mathbb{1}_{y \neq y'} \, \mathrm{d}\rho(z) \, \mathrm{d}\rho(z')$, SGD optimizer.



(d) Functional: $F(\rho) = \mathrm{OTDD}(\mathrm{D}_\rho, \mathrm{D}_\beta) + \lambda \iint -\|\mathbf{x} - \mathbf{x}'\|^2 \mathbb{1}_{y \neq y'} \, \mathrm{d}\rho(z) \, \mathrm{d}\rho(z')$, SGD optimizer.

Figure 8. Gradient flows starting from dataset ρ_0 (red) advecting towards β (blue) driven by different functionals, using `SGD+jd-vl` dynamics in all cases.

E. Additional Experimental Results on *NIST Flows

In Figure 9 we show the effect of the transformation h_θ parametrized as a neural network and learnt from data to mimic the effect of the flow mapping $h_{\text{flow}} : \mathbf{x}_0 \mapsto \mathbf{x}_T$.

Figure 9. **Left**: initial particles $\mathbf{x}_0$ taken from USPS. **Center:** intermediate state of particles $\mathbf{x}_t$ after gradient flow driven by similarity-seeking functional $\mathcal{T}_\beta(\rho) = \text{OTDD}(\mathrm{D}_\rho, \mathrm{D}_\beta)$ for D_β :MNIST. **Right**: particles mapped by using a parametric approximation of h_{flow} learnt from data.

As described in Section 7.2, we use gradient flows to approach to problem of transfer learning. Figure 10 shows results on the 5- and 10-shot tasks on the NIST datasets. Notably, the results follow a similar trend as Figure 3e, although, as expected by the smaller target datasets, the classification errors are higher.

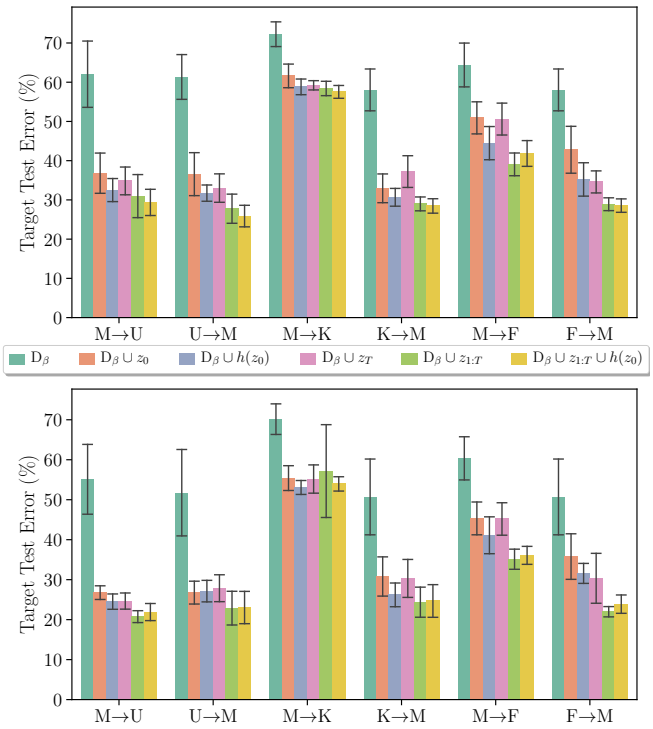


Figure 10. Transfer learning results on image datasets for 5-shot (top) and 10-shot (bottom) tasks. See Section 7.2 for symbol key.

F. Additional Miscellaneous Experimental Results

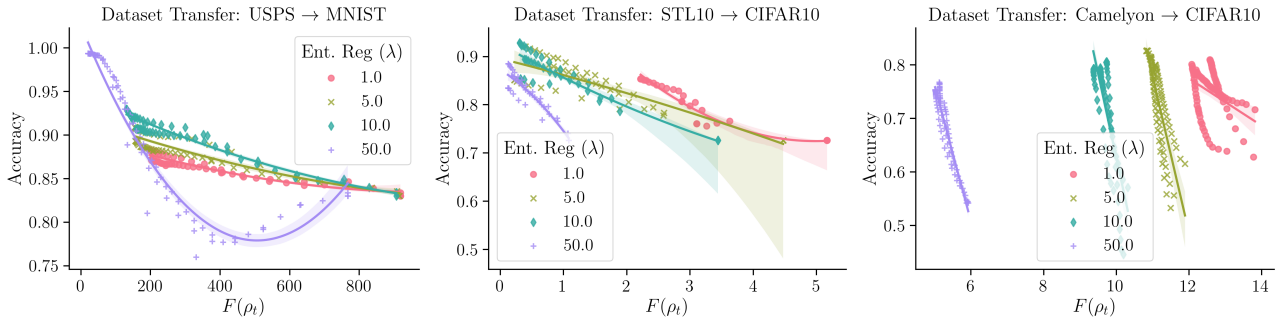


Figure 11. **Qualitative evaluation of flows**: for the same setting as in Figure 6, we plot here the flow objective against the accuracy. The strong correlation between these quantities shows that the OTDD functional provides a good proxy for dataset transfer quality. The relation between these quantities is mostly linear and monotonic, except in configurations with an over-regularized OTDD objective, as shown for the $\lambda = 50$ curve in the first plot.

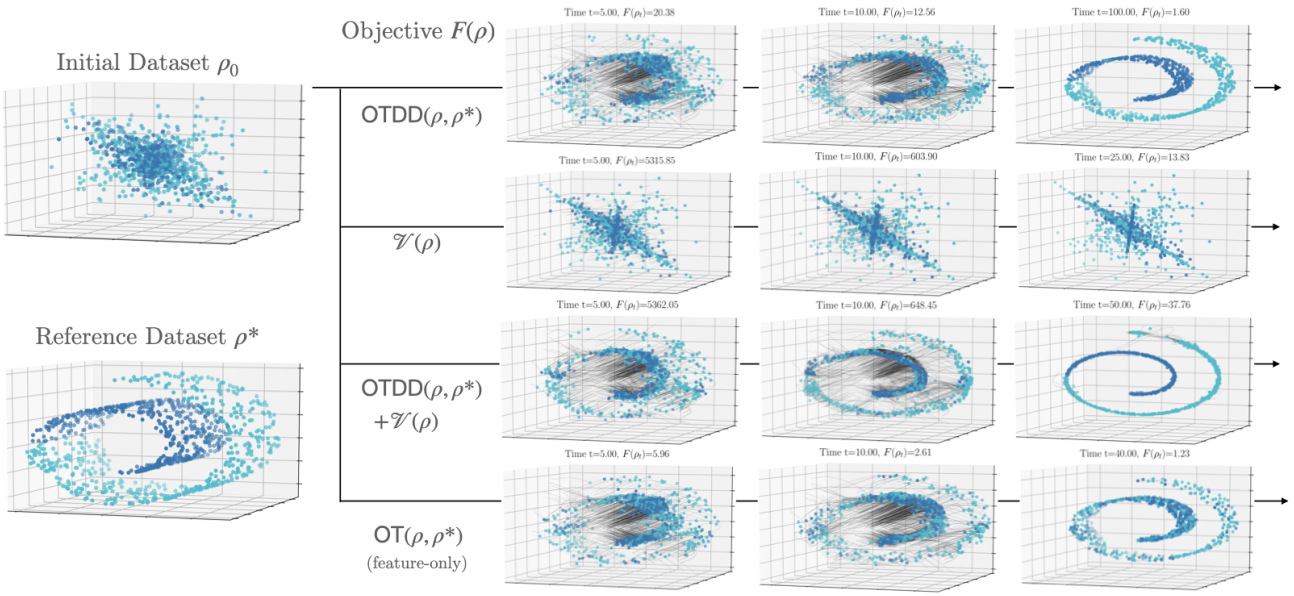


Figure 12. **Shaping datasets via flows**: our framework allows for simple and principled transformation of classification datasets by following the gradient flow of a functional objective, such as: similarity to a reference dataset (shown here as OTDD$(\cdot, \rho^*)$), a function enforcing collapse along a given dimension (shown here as $\mathcal{V}(\rho)$), or a combination thereof. In the bottom row we show the flow obtained with a vanilla (feature-only) optimal transport distance functional, which unsurpisingly misses the important class structure of the dataset.