Distribution-Conditioned Transport

Anonymous Authors[1]

Abstract

Learning a transport model that maps a source distribution to a target distribution is a canonical problem in machine learning, but scientific applications increasingly require models that can generalize to source and target distributions unseen during training. We introduce distribution-conditioned transport (DCT), a framework that conditions transport maps on learned embeddings of source and target distributions, enabling generalization to unseen distribution pairs. DCT also enables semi-supervised learning for distributional forecasting problems: because it learns from arbitrary distribution pairs, it can leverage distributions observed at only one timepoint to improve transport prediction. DCT is agnostic to the underlying transport mechanism, supporting models ranging from flow matching to distributional divergence-based models (e.g. Wasserstein, MMD). We demonstrate the practical performance benefits of DCT on synthetic benchmarks and four applications in biology: batch effect transfer in single-cell genomics, perturbation prediction from mass cytometry data, learning clonal transcriptional dynamics in hematopoiesis, and modeling T-cell receptor sequence evolution.

1. Introduction

Learning transport maps between probability distributions is a central challenge across machine learning and the sciences. A wide variety of approaches have addressed the one-to-one transport problem: learning a map that pushes a source density P_0 to a target density P_1 (Goodfellow et al., 2014; Rezende & Mohamed, 2015; Genevay et al., 2018; Liu et al., 2022; Lipman et al., 2023; Albergo et al., 2023b). However, a new class of problems is emerging where the goal is not only to model the evolution between a single pair of distributions, but rather to model dynamics in a way that can generalize across source and target distributions.

A concrete motivating example of this shift can be seen in the study of cellular dynamics. Experimental techniques such as clonal lineage-traced genomics have enabled the generation of datasets containing snapshots not just of a single cell population, but of thousands of distinct populations (clones) evolving in parallel (Biddy et al., 2018; Weinreb et al., 2020; Wagner & Klein, 2020). These datasets can be sparse, in the sense that not all time-marginals are observed for all populations. For example, we may observe some populations at both an initial and final times t_0 and t_1, while others are "orphans" observed only at either t_0 or t_1.

Two lines of literature have begun to develop tools for these settings. Multimarginal stochastic interpolants offer an approach to learn dynamics between any pair of a fixed set of K distributions, solving a K-to-K transport problem (Albergo et al., 2023a), but cannot condition on a continuous space of distributions nor generalize to distributions unseen during training. In another direction, Fishman et al. (2025) develop approaches to learn "autoencoders" on the space of distributions by simultaneously learning to embed distributions and sample from the distribution conditional on that embedding. Meta flow matching (Atanackovic et al., 2024) is an early approach that couples a population encoder with a flow-matching transport map for generalization across source populations. While effective in settings where many source-target pairs are available (e.g., if we observed many pairs of clonal lineages at both t_0 and t_1), the MFM framework cannot ingest unpaired marginal distributions (e.g., cell populations observed at a single timepoint).

In this work, we unify these perspectives. We first generalize and formalize the approach in Atanackovic et al. (2024), demonstrating how distribution encoders developed in Fishman et al. (2025) can be coupled with a broad class of transport models for source-conditioned transport. This immediately enables us to handle any-to-any transport by conditioning on both source and target distribution embeddings. This formulation enables us to generalize across distributions, predicting transport maps between population pairs unseen during training, as well as allowing us to make use of unstructured, partial observations (such as orphan marginals from incomplete time-series). We demonstrate

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

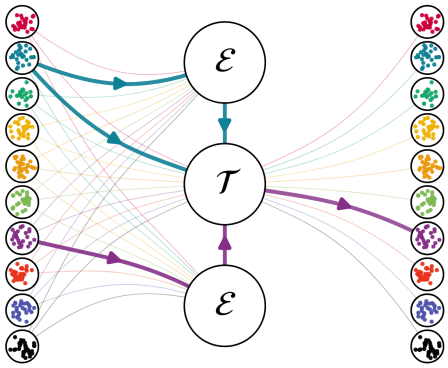

Figure 1. The distribution-conditioned transport framework. A source distribution (teal) is pushed to a target distribution (purple) by a transport model $\mathcal{T}$ which is conditioned on distribution embeddings learned by an encoder $\mathcal{E}$. The learned transport map is universal in the sense that any distribution can in principle be pushed to any distribution by conditioning on the corresponding source and target embeddings.

the effectiveness of our approach first using a set of synthetic benchmark datasets, then on real-world applications in biology. The codebase is available here.

2. Problem setting

Modern datasets in the sciences increasingly have multiscale structure. An example of this can be seen in single-cell RNA-sequencing (scRNA-seq) data. Rather than making measurements of cells from a single experimental condition, modern datasets increasingly consist of cells from many distinct donors (Yazar et al., 2022), timepoints (Qiu et al., 2024), perturbations (Zapatero et al., 2023), or clones (Weinreb et al., 2020), each of which can be thought of as inducing its own distribution over cell states.

This multiscale structure can be formalized as a hierarchical model. Suppose we have sets of samples (e.g., cell gene expression profiles) from n different conditions (e.g., donors). For each condition i, we observe a set of m samples $S_i = \{x_{ij}\}_{j=1}^{m}$, where each sample is a vector $x_{ij} \in \mathcal{X}$. Each condition induces a distribution P_i, and we can think of these condition-specific distributions as being drawn from a shared metadistribution Q over the space of probability measures $\mathcal{P}(\mathcal{X})$: $x_{ij} \sim P_i$ and $P_i \sim Q$. The hierarchical dataset consists of these n sample sets: $\mathcal{D} = \{S_i\}_{i=1}^{n}$.

Many practical challenges in analyzing these hierarchical datasets can be framed as transport problems. For example, in scRNA-seq data, one often is interested in how a distribution would appear under different technical effects (batch integration), how distributions change in response to unseen treatment conditions (perturbation prediction), or forecasting how distributions evolve (dynamic inference).

While a wide variety of tools exist to model transport between a single pair of distributions, these hierarchical datasets and distributional tasks require transport models which can generalize across a broad range of distributions. In this work, we introduce *distribution-conditioned transport*, a framework for solving transport problems on the space of distributions. Concretely, this framework will allow us to solve three classes of practically relevant problems:

Supervised (one-to-one) transport between prescribed pairs of source and target distributions. For example, in clonal lineage-traced scRNA-seq data we observe many clonal populations at multiple timepoints. Each clone has a pair of population-level observations $(P_i^{t=1}, P_i^{t=2})$, and the task of interest is to forecast $P_i^{t=2}$ given $P_i^{t=1}$.

Unsupervised (any-to-any) transport between any pair of distributions. In some cases, we may want to transport between any pair of distributions at test time, with no particular source-target coupling. For example, in scRNA-seq batch integration, we wish to transport cells between any pair of experimental batches, enabling any-to-any batch correction even for experimental batches unseen during training.

Semi-supervised transport with access to orphan marginals. For example, real lineage tracing data often contains incomplete time-series with many populations observed at only a single timepoint (Weinreb et al., 2020). Our goal is still to forecast $P_i^{t=2}$ for any $P_i^{t=1}$ while making use of these orphan marginals.

In the following section, we will show how distribution-conditioned transport provides a framework for addressing each of these problem settings. First, we will show that supervised transport problems can be solved by source-conditioned transport models, formalizing and generalizing the approach of Meta Flow Matching (Atanackovic et al., 2024). Then, we will introduce source-target-conditioned transport models for the unsupervised transport setting, which subsumes the setting addressed by multimarginal stochastic interpolants (Albergo et al., 2023a). Finally, we will show how these two perspectives can be combined to solve the semi-supervised transport problem when one has access to sparse observations of population pairs.

3. Methods

3.1. Distribution encoders

We follow Fishman et al. (2025), defining a *distribution encoder* $\mathcal{E} : S_i \mapsto z_i \in \mathbb{R}^d$, which produces a fixed-dimensional embedding z_i intended to summarize the entire distribution P_i, rather than any particular cell. The key aim of distribution encoders is that z_i reflects only the underlying distributional signal and not sampling noise in S_i. To enforce this the encoder must be *distributionally invariant*:

(i) *permutation invariant*, so that reordering samples does not change $\mathcal{E}(S_i)$, and (ii) *proportionally invariant*, so that uniformly duplicating samples does not change $\mathcal{E}(S_i)$.

When these hold, the encoder depends only on the empirical measure

$$\widehat{P}_i \;=\; \frac{1}{m_i}\sum\nolimits_{j=1}^{m_i}\delta_{x_{ij}}.$$

In particular, there exists a measurable functional ϕ such that $\mathcal{E}(S_i) = \phi(\widehat{P}_i)$. We refer to $z_i = \phi(\widehat{P}_{i,m})$ as a *distribution embedding*. A key property of such encoders is that they admit a central limit theorem (CLT). Let

$$z_i^\star = \lim_{m_i\to\infty}\phi\Big(\widehat{P}_i\Big) = \phi(P_i)$$

denote the population-level embedding obtained in the infinite-sample limit. Under the invariances above and Hadamard differentiability of ϕ, we have

$$\sqrt{m_i}\,\big(\mathcal{E}(S_i) - z_i^\star\big) \;\xrightarrow{\;d\;}\; \mathcal{N}(0,\Sigma_i) \tag{1}$$

for a covariance Σ_i depending on P_i and the encoder $\mathcal{E}$.

For our purposes, this CLT is the most important feature of distribution encoders because, as proved in (Fishman et al., 2025), it implies that for any sufficiently smooth downstream objective we can train using moderate-sized subsamples and recover population-level behavior up to $O(m^{-1/2})$ error. We formalize this plug-in limit theory for DCT in Prop. 3.1 and App. A.

3.2. Supervised (one-to-one) transport

A key setting for distributional transport is when we observe "pairs" of distributions. For example, in perturbation prediction, we observe cell populations of various types (e.g., cell lines, donors) in their untreated and perturbed states. Here, the goal is to transport unperturbed cell populations into their perturbed condition. This transport setting is analogous to supervised learning: we observe many source sets of cells and their corresponding target distribution, and we want to learn a one-to-one map from the source to the target. We show that these supervised problems can be addressed by conditioning transport maps on embeddings of the source distribution. We refer to these as source-conditioned transport models.

A number of recent works have developed particular instances source-conditioned models such as (Atanackovic et al., 2024; Klein et al., 2025) in the context of flow matching or (He et al., 2025) using a direct MMD objective. Our notion of source-conditioning generalizes these under a common framework and clarifies the underlying assumptions under which they enable distributional transport.

Formally, we will assume throughout this section that we have a distribution of pairs sampled from some meta distribution (e.g., control cell populations and their corresponding perturbed populations)

$$(P_1^{\mathrm{src}}, P_1^{\mathrm{tgt}}), \ldots (P_n^{\mathrm{src}}, P_n^{\mathrm{tgt}}) \sim Q_{\mathrm{joint}},$$

and we observe empirical sets

$$S_i^{\mathrm{src}} = \{x_{ij}^{\mathrm{src}}\}_{j=1}^{m_i^{\mathrm{src}}} \sim P_i^{\mathrm{src}} \text{ and } S_i^{\mathrm{tgt}} = \{x_{ij}^{\mathrm{tgt}}\}_{j=1}^{m_i^{\mathrm{tgt}}} \sim P_i^{\mathrm{tgt}}.$$

We encode the *source only*, so $z_i^{\mathrm{src}} = \mathcal{E}\left(S_i^{\mathrm{src}}\right)$, where $\mathcal{E}$ is a distribution encoder as laid out in Sec. 3.1. We can then learn a source-conditioned transport map acting on individual samples,

$$\mathcal{T}:\; \mathcal{X}\times\mathbb{R}^d \to \mathcal{X}, \quad x_i^{\mathrm{src}} \mapsto \hat{x}_i^{\mathrm{tgt}} = \mathcal{T}\big(x_i^{\mathrm{src}} \mid z_{\mathrm{src}}\big),$$

so that transported samples asymptotically follow the target distribution:

$$\mathcal{T}(S_i^{\mathrm{src}} \mid \mathcal{E}(S_i^{\mathrm{src}})) \;\xrightarrow[m_i^{\mathrm{src}}\to\infty]{d}\; P_i^{\mathrm{tgt}} \tag{2}$$

Here $\mathcal{T}$ can be any conditional distributional transport model, and under a supervised pairing policy, the correct destination P_i^{tgt} is implied once the source is specified.

We jointly fit $\mathcal{E}$ and $\mathcal{T}$ using the native loss of the chosen transport mechanism, written abstractly as

$$\mathcal{L}_{sc} = \ell\big(S_i^{\mathrm{tgt}},\, \mathcal{T}(S_i^{\mathrm{src}} \mid \mathcal{E}(S_i^{\mathrm{src}}))\big)\,, \tag{3}$$

where ℓ may be a flow-matching objective (Atanackovic et al., 2024), a distributional divergence (e.g. Sinkhorn/MMD) (He et al., 2025), or any other generative transport model (see Sec 4.2). Because plug-in losses are mean consistent and inherit CLTs (Prop. 3.1 and App. A; see also Cor. A.6), we train with minibatches from S_{src} and S_{tgt} without biasing the population-level objective in the large-m limit (Alg. 1). After training, $\mathcal{T}(\cdot \mid \mathcal{E}(\cdot))$ can be applied to unseen source samples to generate counterfactual realizations under their implied targets.

Our notion of a *transport map* (see Sec. 4.2) is intentionally broad and includes models that, in some regimes, behave more like conditional generation than a coupling-selecting pushforward. In particular, conditional generative models with access to additional noise can achieve Eqn. 2 and match the target distributions while ignoring the source sample x_i^{src} they are supposed to condition on. We prove this in App. A.2 and provide a practical alignment diagnostic to detect this failure mode (Tab. 5). Importantly, this is a property of the underlying mechanism/objective rather than the DCT conditioning framework: DCT enables models to condition on distributions, but does not impose a within-distribution coupling structure beyond what the transport map would learn between a single pair of distributions.

3.3. Unsupervised (any-to-any) transport

In some settings, we wish to transport not between specific pairs of source-target distributions, but rather any pair of distributions. For example, in batch integration for scRNA-seq data, we are interested in the counterfactual of what one

dataset would look like under a different experimental batch condition – and this counterfactual is of interest for any pair of experimental batches.

In other settings, we are interested in particular source-target pairings, but we do not have access to many true pairs. Returning to our running lineage tracing case, in many datasets, we may only observe the same clone in both timepoints a small fraction of the time. In Weinreb et al. (2020), only about 10% of clones are observed at multiple timepoints. Here, we would ideally like a model to also make use of the abundance of unpaired data.

To address both of these settings, we develop an "unsupervised" analogue to the "supervised" transport setting considered above. In this unsupervised setting, we simply have a set of unstructured distributions: $P_1, \dots P_n \sim Q$ and associated samples: $S_i = \{x_{ij}\}_{j=1}^{m_i} \sim P_i$.

Our goal is to develop a model capable of learning to transport between any two distributions i and i'. Multimarginal stochastic interpolants (Albergo et al., 2023a) developed a class of models that can flow between any pair of distributions from a fixed initial set of K distributions by effectively embedding each distribution as a corner of the K-simplex and conditioning on the source and target corners. Our notion of source-target conditioned transport models leverages distribution encoders to learn distribution embeddings, enabling us to generalize to a new $(K+1)^{\text{th}}$ distribution, and to embed a continuous number of distributions.

$$\mathcal{T}: \mathcal{X} \times \mathbb{R}^d \times \mathbb{R}^d \to \mathcal{X},\ x_i \mapsto \widehat{x}_{i\to i'} = \mathcal{T}\big(x_i \mid z_i, z_{i'}\big),$$

to satisfy

$$\mathcal{T}(S_i \mid \mathcal{E}(S_i), \mathcal{E}(S_{i'})) \xrightarrow[m_i, m_{i'} \to \infty]{d} P_{i'} \tag{4}$$

Then, using the same plug-in loss logic (Prop. 3.1), we can train on samples using the objective conditioning on the encoder for the transport map of choice

$$\mathcal{L}_{\text{stc}} = \ell\left(S_{i'}, \mathcal{T}(S_i \mid \mathcal{E}(S_i), \mathcal{E}(S_{i'}))\right) \tag{5}$$

An important point is that our Alg. 1 does not incur any quadratic cost *per gradient step*, even when training on all pairs of distributions. The computational efficiency is the same as the base transport model, up to a forward pass of the distribution encoder.

3.4. Semi-supervised transport

An important use of the unsupervised transport modelling objective is in partial supervision: we want to use all available data to learn a high-quality transport model, but at test time, we are concerned with prediction of a specific target distribution for each source. Source-target-conditioned models can be applied in this semi-supervised setting through two practical adaptations: (1) latent predictions of target distribution embeddings and (2) sampling distribution pairs during training (i.e., choosing Q_{joint}) in a way that aligns with the supervised task at hand.

Algorithm 1 Training distribution-conditioned models

Input: Dataset $\mathcal{D}$, loss ℓ, metadistribution Q_{joint}
for each training step **do**
  Sample indices $(u, v) \sim Q_{\text{joint}}$
  Subsample $\hat{S}_u \subset S_u, \hat{S}_v \subset S_v$ // justified by Prop. 3.1
  // embed distributions and compute loss

source-conditioned	*source-target-conditioned*
$z_u \leftarrow \mathcal{E}(\hat{S}_u)$	$z_u, z_v \leftarrow \mathcal{E}(\hat{S}_u), \mathcal{E}(\hat{S}_v)$
$\mathcal{L} \leftarrow \ell(\hat{S}_v, \mathcal{T}(\hat{S}_u \mid z_u))$	$\mathcal{L} \leftarrow \ell(\hat{S}_v, \mathcal{T}(\hat{S}_u \mid z_u, z_v))$

  Backpropagate $\mathcal{L}$
end for

To make this concrete, we return to the lineage tracing case where we want to leverage all the clones, but the test-time objective is still clonal fate prediction, as in the supervised setting. To convert the any-to-any model into a "supervised" model we can fit a lightweight (e.g., linear) model to predict $z_{\text{src}} \mapsto z_{\text{tgt}}$ using the subset of available (src, tgt) pairs for the task of interest; we then evaluate $\mathcal{T}(\cdot \mid z_{\text{src}}, \widehat{z}_{\text{tgt}})$ to generate counterfactual samples "as if" drawn from P_{tgt} in a semi-supervised manner.

While the source-target conditioned model provides an interface for any-to-any transport, we need not train on all possible pairs. In some scenarios, only certain pairs are of interest. For example, in lineage tracing, it makes more sense to learn forward-time transport between clones ($t_1 \to t_2$) rather than all pairs including potentially uninformative within-timepoint and reverse-time transport. In general, we can define a meta-distribution Q_{joint} over source-target pairs, allowing practitioners to encode domain-specific structure into the model.

Proposition 3.1: Loss CLT (informal)

Fix indices (u, v). Let $z_u = \mathcal{E}(\hat{S}_u)$ and $z_v = \mathcal{E}(\hat{S}_v)$ be embeddings from minibatches of sizes m and m', with limits $z_u^\star = \phi(P_u)$ and $z_v^\star = \phi(P_v)$. Under Fishman et al. (2025) and the sampling schemes in App. A,

$$\left(\sqrt{m}(z_u - z_u^\star),\ \sqrt{m'}(z_v - z_v^\star)\right) \overset{d}{\Longrightarrow} \mathcal{N}\big(0, \Sigma_{uv}\big),$$

where $\Sigma_{uv} = \text{diag}(\Sigma_u, \Sigma_v)$ for independent minibatches (product coupling) and may have off-diagonal blocks for paired/coupled minibatches. Consequently, losses are mean consistent and admit plug-in CLTs; see App. A.

4. Related work

4.1. Distribution embeddings

Distribution conditioned transport relies on a vector representation of a distribution, or a distribution embedding.

Kernel methods, including kernel mean embeddings, provide a toolkit for representing probability measures as points in a reproducing kernel Hilbert space (Smola et al., 2007; Muandet et al., 2012; Oliva et al., 2013; Szabo et al., 2015; Muandet et al., 2017). In parallel, others have explored learning distribution embeddings which preserve Wasserstein distances between distributions (Haviv et al., 2024), in analogy to multidimensional scaling. Our distribution encoders follows the formal framework in Fishman et al. (2025) to learn distributionally-invariant embeddings by leveraging permutation-invariant architectures with a particular class of pooling operators (Zaheer et al., 2017; Wagstaff et al., 2021; Zhang et al., 2022).

4.2. Transport models

A range of generative transport mechanisms can be used as the conditional map $\mathcal{T}(\cdot \mid z)$ or $\mathcal{T}(\cdot \mid z, z')$ in our framework. Classical adversarial approaches learn source–target mappings via discriminators, including GANs (Goodfellow et al., 2014) and Wasserstein GANs (Arjovsky et al., 2017). We can also transport distributions by minimizing distributional divergences (Gneiting & Raftery, 2007; Li et al., 2015; Sutherland et al., 2016; Genevay et al., 2018; Kolouri et al., 2019). Normalizing flows provide invertible transport parameterizations through continuous change-of-variable models (Rezende & Mohamed, 2015). Continuous-time flow-based models, such as flow matching, rectified flows, and stochastic interpolants (Lipman et al., 2023; Liu et al., 2022; Albergo et al., 2023b), define transport by learning a velocity field along a path interpolating between source and target.

On discrete sequences, unsupervised machine translation methods (Lample et al., 2018a; Artetxe et al., 2018; Lample et al., 2018b) learn bidirectional maps between languages using only monolingual corpora, providing a discrete analogue of our unpaired any-to-any transport between distributions. More recently, flow-matching has been extended to the simplex and general discrete domains (Gat & Lipman, 2024; Cheng et al., 2024; Davis et al., 2024).

All of these models define mappings that take samples from a source distribution and produce samples matching a target distribution. Our framework is orthogonal to the specific transport mechanism: any such model can be conditioned on distribution embeddings to yield a source-conditioned or source–target–conditioned transport map that satisfies the population-level limits in Eqn. (2) and Eqn. (4).

4.3. Multi-distribution transport

Meta Flow Matching (Atanackovic et al., 2024) develops the idea of source-conditioning for learning transport maps across contexts. Our notion of source-conditioned models generalizes their approach to a broader class of transport models and demonstrates the conditions under which we can expect it to achieve Eqn. 2. A number of recent cellular perturbation response prediction methods are also effectively source-conditioned models (Klein et al., 2025; He et al., 2025; Adduri et al., 2025).

Multimarginal stochastic interpolants (MMSI) studies a form of any-to-any transport between a fixed set of K distributions (Albergo et al., 2023a). This is a particular instance of the unsupervised modelling problem we consider, where Q is chosen to cover a discrete set of distributions that are all seen at training time. Additionally, MMSI enforces that transport paths go through all K distributions to learn a barycenter, but in DCT we do not impose any constraint.

Style-transfer and unpaired image-translation methods provide a complementary line of work. Approaches such as CycleGAN, StarGAN, and MUNIT (Zhu et al., 2017; Choi et al., 2018; Huang et al., 2018) condition the generator on a domain identifier, typically a one-hot categorical label or an instance-level style code. These approaches are restricted to a fixed, finite set of domains and cannot generalize transport to a new domain without retraining.

4.4. Transport models for biological data

The family of transport models described above are widely used in biology. They have attracted significant recent interest for at least two problems: predicting cellular perturbation responses from scRNA-seq, imaging, and mass cytometry data (Atanackovic et al., 2024; Klein et al., 2025; Adduri et al., 2025; He et al., 2025; Bunne et al., 2023), and inferring cellular dynamics from snapshot observations (Schiebinger et al., 2019; Lavenant et al., 2021; Tong et al., 2023a;c; Vinyard et al., 2025).

5. Unsupervised Transport

Using both synthetic data and real scRNA-seq data, we will show that source-target-conditioned DCTs can address the any-to-any unsupervised transport problem in practice. We will compare source-target-conditioned DCTs using a mean pooled deep-sets encoder (see. App. C.2) to K-to-K models implemented using a "one-hot" $\mathcal{E}$ rather than a distributionally invariant architecture. Using the FM transport map, this is effectively a MMSI model (Albergo et al., 2023a), with minor differences we discuss in App. B. To enable test-time generalization, K-to-K models are conditioned on the most similar training set distributions, determined by nearest-neighbor over distribution centroids. We implement both approaches across sliced Wasserstein distance (SWD) models, Energy score models, and flow matching (FM) models (see App. C), and evaluate using MMD in the main text with SWD and Energy distances available for all experiments in the appendix (D).

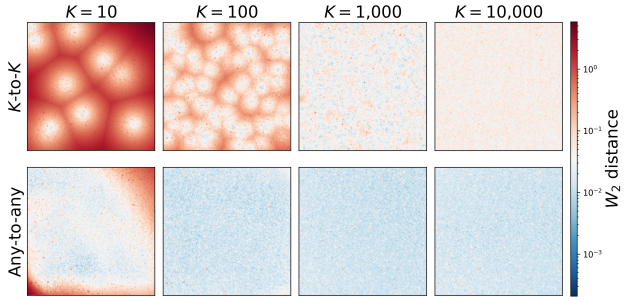

Figure 2. Bivariate normal transport error landscape. K-to-K (top) vs. any-to-any (bottom), showing W_2 distance for targets $\mu \in [0, 5]^2$ across K. The K-to-K model predicts via nearest training distribution, yielding Voronoi-like error patterns; the any-to-any model embeds targets directly and achieves uniformly lower error across all K.

5.1. Gaussian transport

We first study transport between simple distributions in two dimensions, where we can control the data-generating process and directly visualize performance.

We construct datasets of bivariate normal (MVN) distributions and Gaussian mixture models (GMM). For MVN, we sample parameters (μ_i, Σ_i), $i = 1, \ldots, n$, with means $\mu_i \in [0, 5]^2$ drawn from a uniform prior and covariances $\Sigma_i \in \mathbb{R}^{2\times 2}$ drawn from an inverse–Wishart prior. For GMM, we additionally sample mixture weights from a Dirichlet prior and per-component parameters (see App. E.1 for details). We choose $K \in \{10, 100, 1{,}000, 10{,}000\}$ distinct parameter sets and draw $n = 50{,}000$ sample sets.

For each K we train two types of model. The first is a conventional *K-to-K* architecture that treats each of the K distinct distributions as a discrete label (a "corner") and learns transport conditional on distribution-specific labels. At test time, this model cannot condition on a new target distribution, so we assign each target to its nearest training distribution. The second is our *source–target conditioned* model from Sec. 3.3, which encodes both source and target sets via the distribution encoder $z_i = \mathcal{E}(S_i)$ and learns a transport map $\mathcal{T}(x \mid z_{\text{src}}, z_{\text{tgt}})$ that can operate between arbitrary pairs of distributions, including unseen targets.

To visualize zero-shot performance, we fix a single source distribution from the training set and evaluate transport to a dense grid of target distributions, with $\mu \in [0, 5]^2$ and corresponding random covariances. Results (Figs. 2 and 3) show that the K-to-K model outperforms DCT on in-distribution targets when K is small, but loses this advantage as K grows. And crucially, on out-of-distribution targets, DCT achieves significantly lower error: the baseline cannot extrapolate beyond its training corners, while the any-to-any model generalizes smoothly. This pattern holds not only for MVN but also GMM, suggesting that the benefits of DCT extend to complex distributions.

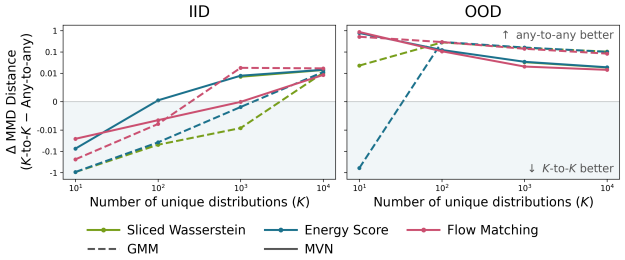

Figure 3. Unsupervised transport model generalization. The gap between K-to-K based embedding and any-to-any distribution encoders across generator families, evaluated on in-distribution (IID) and out-of-distribution (OOD) test sets. Positive values indicate the distribution encoder achieves lower transport cost. At low K, the embedding encoder outperforms on IID data while the distribution encoder shows stronger OOD generalization.

Generator	K-to-K	Any-to-any
SWD	$0.3411_{\pm 0.0351}$	$0.1167_{\pm 0.0119}$
Energy	$0.3623_{\pm 0.0327}$	$\mathbf{0.0729}_{\pm 0.0090}$
FM	$0.4235_{\pm 0.0540}$	$0.1480_{\pm 0.0157}$
scVI	$0.8056_{\pm 0.0643}$	

Table 1. Batch effect transfer quality on held-out samples. MMD distance (↓) averaged across all pairs of 3 held-out donors.

5.2. Transferring scRNA-seq batch effects

A common problem in scRNA-seq analysis is technical variation between experimental batches (Luecken et al., 2022). We show that source-target-conditioned models can be used to push cells from a source batch to a target batch condition, making a prediction for how cells appear under different technical effects. This problem, which we refer to as batch effect transfer, is closely related to batch integration.

Source-target-conditioning, in principle, allows us to fit a single batch transfer model which can be applied to unseen experimental batches at test time. We demonstrate this using a real-world dataset of murine pancreas cells from 56 mice across ages and growth conditions (Hrovatin et al., 2023). We hold out all mice from a single condition (old age), inducing a distribution shift.

We might expect that the K-to-K models are not be able to adapt to distribution shift, given that they can only condition on distribution identities seen at test time, while DCTs in principle may be able to encode information about the correct target distribution.

DCTs outperform K-to-K alternatives and scVI (Lopez et al., 2018) on transport of cells from held-out experimental batches across all tested generator families (Table 1, implementation details in App. G). In Fig. 4, we visualize model predictions for batch effect transfer, which similarly show that DCT predictions more closely match the target cell distribution than those of a K-to-K model.

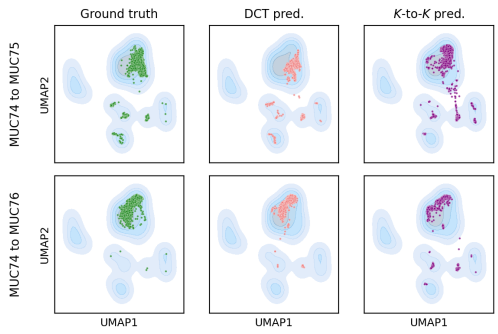

Figure 4. Batch effect transfer for two held-out donor pairs (rows). Blue contours represent the reference density ($\sim 3 \cdot 10^5$ cells) computed from all 56 donors. (*left*) Ground truth cells from the target donor. (*center*) predictions from DCT and (*right*) K-to-K model. DCT predictions more closely match ground truth than K-to-K predictions.

6. Semi-supervised Transport

Across synthetic data, mass cytometry data, and protein sequence data, we will show that source-target-conditioned DCTs can improve performance in the semi-supervised setting. We compare source-target-conditioned DCTs with their source-conditioned counterparts, across SWD-based, Energy-based, and FM transport maps (see App. C). We evaluate using three distributional metrics (MMD, SWD, and Energy) with MMD presented in the main text and all three available in the App. D.

The source-conditioned models we evaluate against include the source-conditioned FM, which is identical to MFM (Atanackovic et al., 2024) aside from the choice of a CLT-satisfying population encoder (see App. B for discussion). All semi-supervised approaches use ridge regression fit on the same data as the supervised (SC) model, with the regularization selected through cross-validation. We also compare with "oracle" models which are STC models allowed to condition on the embedding of the true distribution. A central finding here is that in real data applications we can often improve OOD generalization of supervised models by incorporating a wider set of data without explicit paired distributions (see the orphan marginals discussion in Sec. 3.4).

6.1. Gaussian Transport

Returning to our Gaussian setting, we build a benchmark for semi-supervised learning. We define paired distributions via fixed transformations: for MVN, a simple shift $Y = X + b$; for GMM, a bimodal target with off-axis displacements. We compare three approaches: (i) a *supervised* source-conditioned model trained on paired examples with means $\|\mu\|_\infty \leq 2.5$, (ii) the *semi-supervised* any-to-any model from Sec. 5.1, and (iii) an *oracle* using the true target

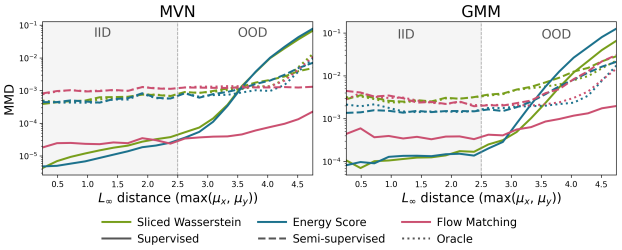

Figure 5. Semi-supervised transport model generalization. Supervised models are trained on distributions with $\|\mu\|_\infty \leq 2.5$ (shaded) and evaluated across $\|\mu\|_\infty \leq 5$. Supervised models (red) degrade sharply outside the training support (except flow matching models), while semi-supervised models (blue) maintain stable performance approaching the oracle (green). Results shown for multivariate normal (left) and Gaussian mixture (right).

embedding. All models are evaluated across $\|\mu\|_\infty \leq 5$, testing generalization beyond the supervised training support.

Figure 5 shows energy distance as a function of L^∞ distance for both settings. Within the training support, supervised and semi-supervised methods achieve comparable performance. Beyond it, the semi-supervised approach maintains substantially lower error: the any-to-any objective learns distributional structure that enables extrapolation even when the latent predictor must generalize. The flow matching model is an exception here: it generalizes very well outside its support. The extremely strong performance of FM models here may be connected to the tightly coupled transport maps learned by FM (see App. A.2), but this performance is not replicated in any of our real-world experiments.

6.2. Drug perturbation prediction on organoids

We evaluate DCTs on a single-cell mass-cytometry drug screen from Zapatero et al. (2023), which profiles patient-derived organoids (PDOs) from 10 colorectal cancer patients treated with 11 chemotherapy compounds. Each experimental replicate consists of a matched control-treatment pair, yielding 927 population pairs. Cells are characterized by an abundance profile over 43 proteins.

A key challenge is predicting patient-specific treatment response: tumors from different patients exhibit distinct drug sensitivities. The task is to predict the post-treatment distribution given the control population and treatment identity. We evaluate on two splits: (i) held-out replicates within patients seen during training, and (ii) held-out patients, testing generalization.

We compare a source-conditioned baseline (including MFM) against our semi-supervised source-target conditioned model. Both models condition on treatment identity. As shown in Table 2, the source-conditioned model achieves lower error on held-out replicates within seen pa-

		Supervised	Semi-sup.	Oracle
IID	SWD	0.64±.22	0.94±.18	0.10±.00
	Energy	**0.54**±.19	1.03±.32	0.11±.02
	FM	1.01±.23	1.06±.25	0.30±.03
OOD	SWD	2.59±.08	1.97±.22	0.26±.09
	Energy	2.98±.19	**1.91**±.49	0.22±.02
	FM	2.27±.06	2.08±.18	0.53±.13

Table 2. Perturbation prediction perfomance on PDO task. MMD distance $\times 10^2$ (↓) averaged over held-out patients.

	Supervised	Semi-supervised	Oracle
SWD	9.87±.27	8.67±.22	2.99±.05
Energy	9.42±.26	**8.54**±.21	2.88±.05
FM	14.10±.28	9.67±.25	5.08±.10

Table 3. Clonal distribution forecasting on Weinreb et al. (2020). MMD distance (↓) across 628 held-out clones.

tients, but the any-to-any model generalizes better to unseen patients—echoing the IID/OOD tradeoff observed in our Gaussian benchmarks (Fig. 3).

6.3. Modeling clonal population dynamics

We next apply DCTs to learn clonal population dynamics using lineage-traced single-cell RNA-sequencing (scRNA-seq) data from Weinreb et al. (2020). A critical challenge in this dataset is sparsity: there are about $6 \cdot 10^3$ clones measured in total, however only about $2 \cdot 10^3$ of these clones are profiled at multiple timepoints. While source-conditioned approaches can only be trained on clones observed at multiple timepoints, training DCTs with any-to-any pairings allow us to make use of all available marginals. Here, when training without coupling information we restrict source-target pairing to only allow an early timepoint clone as a source distribution and a late timepoint clone as a target distribution.

We compare source-conditioned (supervised) and source-target-conditioned (semi-supervised) approaches using an identical architectures operating on the first 50 principal components of gene expression profiles (implementation details in App. I). As shown in Table 6.3, the source-target-conditioned models significantly outperform the source-conditioned baselines, demonstrating the benefit of incorporating orphan marginals.

6.4. Longitudinal TCR repertoire forecasting

We evaluate DCT on longitudinal T-cell receptor (TCR) repertoire sequencing from COVID-19 patients (Schultheiß et al., 2020). The dataset contains repertoires from 37 patients, of whom only 10 are profiled at multiple timepoints – a sparsity pattern mirroring the orphan marginal regime in lineage tracing. We hold out three longitudinal patients for evaluation and train on the remainder. The task is to forecast a patient's repertoire at time $t{+}1$ given their repertoire at t.

Each repertoire is an empirical distribution over TRB CDR3 amino acid sequences. We embed sequences using a pre-trained ESM backbone (Lin et al., 2023) with mean-pooled DeepSets aggregation. We compare two discrete transport mechanisms: a ProGen-based bridge (Nijkamp et al., 2023) and a discrete flow matching (DFM) bridge (Gat & Lipman, 2024), both trained in the same distribution-conditioned framework. The supervised baseline trains only on within-patient adjacent pairs $(t, t{+}1)$; the semi-supervised model trains on cross-patient pairs (earlier-to-later timepoints), incorporating single-timepoint patients as unpaired marginals.

	Supervised	Semi-supervised	Oracle
Progen	0.0588±.009	0.0587±.009	0.0567±.009
DFM	0.0579±.010	**0.0215**±.004	0.0078±.001

Table 4. TCR repertoire forecasting performance. MMD distance (↓) averaged over held-out patients.

The ProGen model appears degenerate; it learns identical embeddings for all distributions with average cosine similarity 0.998 (see App. J.5). The semi-supervised DFM model reduces energy distance by more than half, demonstrating that cross-patient distributional structure improves forecasting even with limited longitudinal supervision.

7. Conclusion

We introduced *distribution-conditioned transport*, a framework that couples distribution encoders with conditional transport models to learn transport maps which can generalize to unseen source and target distributions. We showed that DCT formalizes and generalizes existing approaches under a unified framework, leading to new models with empirical performance gains on synthetic benchmarks and four real world problems in biology.

Limitations and extensions Although source-target-conditioned models outperform source-conditioned and K-to-K baselines out-of-distribution, they sometimes underperform in-distribution. This may be due to underfitting: while all compared models are trained with equal compute budgets, source-target-conditioned models may have different scaling properties and require more training. In semi-supervised problems, we fit simple linear latent predictors for source-target-conditioned models. It may be fruitful to explore more complex latent prediction schemes.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Adduri, A. K., Gautam, D., Bevilacqua, B., Imran, A., Shah, R., Naghipourfar, M., Teyssier, N., Ilango, R., Nagaraj, S., Dong, M., et al. Predicting cellular responses to perturbation across diverse contexts with state. *bioRxiv*, pp. 2025–06, 2025.

Albergo, M. S., Boffi, N. M., Lindsey, M., and Vanden-Eijnden, E. Multimarginal generative modeling with stochastic interpolants. *arXiv preprint arXiv:2310.03695*, 2023a.

Albergo, M. S., Boffi, N. M., and Vanden-Eijnden, E. Stochastic interpolants: A unifying framework for flows and diffusions. *Journal of Machine Learning Research*, 24:1–63, 2023b.

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein GAN. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pp. 214–223. PMLR, 2017.

Artetxe, M., Labaka, G., Agirre, E., and Cho, K. Unsupervised neural machine translation. In *International Conference on Learning Representations*, 2018.

Atanackovic, L., Zhang, X., Amos, B., Blanchette, M., Lee, L. J., Bengio, Y., Tong, A., and Neklyudov, K. Meta Flow Matching: Integrating Vector Fields on the Wasserstein Manifold. In *The Thirteenth International Conference on Learning Representations*, 4 October 2024.

Biddy, B. A., Kong, W., Kamimoto, K., Guo, C., Waye, S. E., Sun, T., and Morris, S. A. Single-cell mapping of lineage and identity in direct reprogramming. *Nature*, 564(7735):219–224, 2018.

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, 28 November 2023. ISSN 1548-7091,1548-7105. doi: 10.1038/s41592-023-01969-x.

Cheng, X., Chen, T., Li, K., Gao, R., Zhu, J., and Liu, Q. Categorical flow matching. In *International Conference on Learning Representations*, 2024.

Choi, Y., Choi, M., Kim, M., Ha, J.-W., Kim, S., and Choo, J. StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8789–8797, 2018.

Davis, N., Chen, R. T. Q., Hu, Y., and Rezende, D. J. Fisher flow matching: Transport in the probability simplex. In *International Conference on Machine Learning*, 2024.

Fishman, N., Gowri, G., Yin, P., Gootenberg, J., and Abudayyeh, O. Generative distribution embeddings. *arXiv preprint arXiv:2505.18150*, 2025.

Gat, I. and Lipman, Y. Discrete flow matching. In *Advances in Neural Information Processing Systems*, 2024.

Genevay, A., Peyré, G., and Cuturi, M. Learning generative models with sinkhorn divergences. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84, pp. 1608–1617. PMLR, 2018.

Gneiting, T. and Raftery, A. E. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27, 2014.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. A kernel two-sample test. *The journal of machine learning research*, 13(1):723–773, 2012.

Haviv, D., Kunes, R. Z., Dougherty, T., Burdziak, C., Nawy, T., Gilbert, A., and Pe'er, D. Wasserstein Wormhole: Scalable optimal transport distance with transformers. *International Conference on Machine Learning*, 235:17697–17718, 15 April 2024. doi: 10.48550/arXiv.2404.09411.

He, C., Zhang, J., Dahleh, M., and Uhler, C. Morph predicts the single-cell outcome of genetic perturbations across conditions and data modalities. *bioRxiv*, 2025.

Heumos, L., Schaar, A. C., Lance, C., Litinetskaya, A., Drost, F., Zappia, L., Lücken, M. D., Strobl, D. C., Henao, J., Curion, F., Single-cell Best Practices Consortium, Schiller, H. B., and Theis, F. J. Best practices for single-cell analysis across modalities. *Nature reviews. Genetics*, pp. 1–23, 31 March 2023. ISSN 1471-0056,1471-0064. doi: 10.1038/s41576-023-00586-w.

Hrovatin, K., Bastidas-Ponce, A., Bakhti, M., Zappia, L., Büttner, M., Salinno, C., Sterr, M., Böttcher, A., Migliorini, A., Lickert, H., and Theis, F. J. Delineating mouse β-cell identity during lifetime and in diabetes with a single cell atlas. *Nature Metabolism*, 5(9):1615–1637, 11 September 2023. ISSN 2522-5812,2522-5812. doi: 10.1038/s42255-023-00876-x.

Huang, X., Liu, M.-Y., Belongie, S., and Kautz, J. Multimodal unsupervised image-to-image translation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 179–196, 2018.

Klein, D., Fleck, J. S., Bobrovskiy, D., Zimmermann, L., Becker, S., Palma, A., Dony, L., Tejada-Lapuerta, A., Huguet, G., Lin, H.-C., et al. Cellflow enables generative single-cell phenotype modeling with flow matching. *bioRxiv*, pp. 2025–04, 2025.

Kolouri, S., Nadjahi, K., Simsekli, U., Badeau, R., and Rohde, G. Generalized sliced wasserstein distances. *Advances in neural information processing systems*, 32, 2019.

Lample, G., Conneau, A., Denoyer, L., and Ranzato, M. Unsupervised machine translation using monolingual corpora only. In *International Conference on Learning Representations*, 2018a.

Lample, G., Ott, M., Conneau, A., Denoyer, L., and Ranzato, M. Phrase-based & neural unsupervised machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 5039–5049. Association for Computational Linguistics, 2018b.

Lavenant, H., Zhang, S., Kim, Y.-H., and Schiebinger, G. Towards a mathematical theory of trajectory inference. *arXiv [stat.ML]*, (1A):428–500, 18 February 2021.

LeCun, Y., Cortes, C., and Burges, C. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

Li, Y., Swersky, K., and Zemel, R. S. Generative moment matching networks. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pp. 1718–1727. PMLR, 2015.

Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., Smetanin, N., Verkuil, R., Kabeli, O., Shmueli, Y., et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637): 1123–1130, 2023.

Lipman, Y., Chen, R. T. Q., Ben-Hamu, H., Nickel, M., and Le, M. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

Liu, X., Gong, C., and Liu, Q. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Lopez, R., Regier, J., Cole, M. B., Jordan, M. I., and Yosef, N. Deep generative modeling for single-cell transcriptomics. *Nature methods*, 15(12):1053–1058, December 2018. ISSN 1548-7091,1548-7105. doi: 10.1038/s41592-018-0229-2.

Luecken, M. D., Büttner, M., Chaichoompu, K., Danese, A., Interlandi, M., Mueller, M. F., Strobl, D. C., Zappia, L., Dugas, M., Colomé-Tatché, M., and Theis, F. J. Benchmarking atlas-level data integration in single-cell genomics. *Nature Methods*, 19(1):41–50, January 2022. ISSN 1548-7091,1548-7105. doi: 10.1038/s41592-021-01336-8.

Muandet, K., Fukumizu, K., Dinuzzo, F., and Scholkopf, B. Learning from distributions via support measure machines. *Neural Information Processing Systems*, 25:10–18, 29 February 2012.

Muandet, K., Fukumizu, K., Sriperumbudur, B., and Schölkopf, B. Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends® in Machine Learning*, 10(1-2):1–141, 2017. ISSN 1935-8237,1935-8245. doi: 10.1561/2200000060.

Nijkamp, E., Ruffolo, J. A., Weinstein, E. N., Naik, N., and Madani, A. ProGen2: Exploring the boundaries of protein language models. *Cell systems*, 14(11):968–978.e3, 15 November 2023. ISSN 2405-4712,2405-4720. doi: 10.1016/j.cels.2023.10.002.

Oliva, J. B., Póczos, B., and Schneider, J. Distribution to Distribution Regression. *International Conference on Machine Learning*, 28(3):1049–1057, 16 June 2013.

Qiu, C., Martin, B. K., Welsh, I. C., Daza, R. M., Le, T.-M., Huang, X., Nichols, E. K., Taylor, M. L., Fulton, O., O'Day, D. R., Gomes, A. R., Ilcisin, S., Srivatsan, S., Deng, X., Disteche, C. M., Noble, W. S., Hamazaki, N., Moens, C. B., Kimelman, D., Cao, J., Schier, A. F., Spielmann, M., Murray, S. A., Trapnell, C., and Shendure, J. A single-cell time-lapse of mouse prenatal development from gastrula to birth. *Nature*, 626(8001):1084–1093, 14 February 2024. ISSN 0028-0836,1476-4687. doi: 10.1038/s41586-024-07069-w.

Rezende, D. J. and Mohamed, S. Variational inference with normalizing flows. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pp. 1530–1538. PMLR, 2015.

Schiebinger, G., Shu, J., Tabaka, M., Cleary, B., Subramanian, V., Solomon, A., Gould, J., Liu, S., Lin, S., Berube, P., Lee, L., Chen, J., Brumbaugh, J., Rigollet, P., Hochedlinger, K., Jaenisch, R., Regev, A., and Lander, E. S. Optimal-Transport Analysis of Single-Cell Gene Expression Identifies Developmental Trajectories in Reprogramming. *Cell*, 176(4):928–943.e22,

7 February 2019. ISSN 0092-8674,1097-4172. doi: 10.1016/j.cell.2019.01.006.

Schultheiß, C., Paschold, L., Simnica, D., Mohme, M., Willscher, E., von Wenserski, L., Scholz, R., Wieters, I., Dahlke, C., Tolosa, E., et al. Next-generation sequencing of t and b cell receptor repertoires from covid-19 patients showed signatures associated with severity of disease. *Immunity*, 53(2):442–455, 2020.

Smola, A., Gretton, A., Song, L., and Schölkopf, B. A Hilbert space embedding for distributions. In *Lecture Notes in Computer Science*, Lecture notes in computer science, pp. 13–31. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007. ISBN 9783540752240,9783540752257. doi: 10.1007/978-3-540-75225-7_5.

Sutherland, D. J., Tung, H.-Y., Strathmann, H., De, S., Ramdas, A., Smola, A., and Gretton, A. Generative models and model criticism via optimized maximum mean discrepancy. *arXiv preprint arXiv:1611.04488*, 2016.

Szabo, Z., Gretton, A., Poczos, B., and Sriperumbudur, B. Two-stage sampled learning theory on distributions. In *Artificial Intelligence and Statistics*, pp. 948–957. PMLR, 21 February 2015.

Tong, A., Fatras, K., Malkin, N., Huguet, G., Zhang, Y., Rector-Brooks, J., Wolf, G., and Bengio, Y. Improving and generalizing flow-based generative models with minibatch optimal transport. *arXiv [cs.LG]*, 1 February 2023a.

Tong, A., Fatras, K., Malkin, N., Huguet, G., Zhang, Y., Rector-Brooks, J., Wolf, G., and Bengio, Y. Improving and generalizing flow-based generative models with minibatch optimal transport. *arXiv preprint arXiv:2302.00482*, 2023b.

Tong, A., Malkin, N., Fatras, K., Atanackovic, L., Zhang, Y., Huguet, G., Wolf, G., and Bengio, Y. Simulation-free Schrödinger bridges via score and flow matching. *arXiv [cs.LG]*, 7 July 2023c.

Van Der Vaart, A. W. and Wellner, J. A. *Weak convergence*. Springer, 1996.

Vinyard, M. E., Rasmussen, A. W., Li, R., Klein, A. M., Getz, G., and Pinello, L. Learning cell dynamics with neural differential equations. *Nature Machine Intelligence*, 7(12):1969–1984, 18 December 2025. ISSN 2522-5839,2522-5839. doi: 10.1038/s42256-025-01150-3.

Wagner, D. E. and Klein, A. M. Lineage tracing meets single-cell omics: opportunities and challenges. *Nature Reviews Genetics*, 21(7):410–427, 2020.

Wagstaff, E., Fuchs, F., Engelcke, M., Osborne, M. A., and Posner, I. Universal approximation of functions on sets. *Journal of machine learning research: JMLR*, 23(151): 151:1–151:56, 5 July 2021. ISSN 1532-4435,1533-7928.

Weinreb, C., Rodriguez-Fraticelli, A., Camargo, F. D., and Klein, A. M. Lineage tracing on transcriptional landscapes links state to fate during differentiation. *Science*, 367(6479), 14 February 2020. ISSN 0036-8075,1095-9203. doi: 10.1126/science.aaw3381.

Yazar, S., Alquicira-Hernandez, J., Wing, K., Senabouth, A., Gordon, M. G., Andersen, S., Lu, Q., Rowson, A., Taylor, T. R. P., Clarke, L., Maccora, K., Chen, C., Cook, A. L., Ye, C. J., Fairfax, K. A., Hewitt, A. W., and Powell, J. E. Single-cell eQTL mapping identifies cell type-specific genetic control of autoimmune disease. *Science (New York, N.Y.)*, 376(6589):eabf3041, 8 April 2022. ISSN 0036-8075,1095-9203. doi: 10.1126/science.abf3041.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Póczos, B., Salakhutdinov, R., and Smola, A. Deep Sets. *Advances in neural information processing systems*, 30, 10 March 2017.

Zapatero, M. R., Tong, A., Opzoomer, J. W., O'Sullivan, R., Rodriguez, F. C., Sufi, J., Vlckova, P., Nattress, C., Qin, X., Claus, J., et al. Trellis tree-based analysis reveals stromal regulation of patient-derived organoid drug responses. *Cell*, 186(25):5606–5619, 2023.

Zhang, L. H., Tozzo, V., Higgins, J., and Ranganath, R. Set norm and equivariant skip connections: Putting the deep in Deep Sets. *International Conference on Machine Learning*, 162:26559–26574, 23 June 2022. doi: 10.48550/arXiv.2206.11925.

Zhu, J.-Y., Park, T., Isola, P., and Efros, A. A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2223–2232, 2017.

A. Theory

A.1. Why can we train on minibatches?

This appendix formalizes two ingredients used throughout the main text: (i) distribution encoders factor through the empirical measure, and (ii) encoder CLTs yield plug-in CLTs for objectives that condition on the encoder state. The plug-in loss theory is adapted from Fishman et al. (2025, Assump. (C.1)–(C.3)); we include short proofs to make the manuscript self-contained and to handle two DCT-specific points: (a) joint CLTs for source–target conditioned models when the source and target minibatches are *coupled* (paired data), and (b) the small dependence introduced when a loss conditions on a point x that is reused inside the same minibatch used to compute $\mathcal{E}(\hat{S})$. We refer to Fishman et al. (2025); Van Der Vaart & Wellner (1996) for broader context.

A.1.1. Empirical-measure factorization

Let $(\mathcal{X}, \mathcal{B})$ be a measurable space and let $\mathcal{P}(\mathcal{X})$ denote probability measures on it. For $m \in \mathbb{N}$ and $S_m = (x_1, \ldots, x_m) \in \mathcal{X}^m$, write the empirical measure

$$\widehat{P}_m \;:=\; \frac{1}{m}\sum_{j=1}^{m} \delta_{x_j}.$$

These factorization statements mirror the empirical-measure sufficiency/maximal-invariance viewpoint in Fishman et al. (2025, Thm. "Empirical measure is sufficient, minimal, and a maximal invariant"); we include short proofs for completeness and to match DCT notation.

Definition A.1 (Permutation and proportional invariance)**.** Fix $m \in \mathbb{N}$. A map $\mathcal{E}_m : \mathcal{X}^m \to \mathbb{R}^d$ is *permutation invariant* if

$$\mathcal{E}_m(x_1, \ldots, x_m) = \mathcal{E}_m(x_{\pi(1)}, \ldots, x_{\pi(m)}) \qquad \text{for all permutations } \pi \in \mathfrak{S}_m.$$

A family $(\mathcal{E}_m)_{m\geq 1}$ is *proportionally invariant* if for every integer $k \geq 1$ and every $(x_1, \ldots, x_m) \in \mathcal{X}^m$,

$$\mathcal{E}_m(x_1, \ldots, x_m) \;=\; \mathcal{E}_{km}(\underbrace{x_1, \ldots, x_m, \ldots, x_1, \ldots, x_m}_{k \text{ copies}}).$$

Proposition A.2 (Permutation invariance ⇒ factorization)**.** *Fix $m \in \mathbb{N}$ and let $\mathcal{E}_m : \mathcal{X}^m \to \mathbb{R}^d$ be permutation invariant. Then there exists a measurable map ϕ_m defined on the set of empirical measures $\{\widehat{P}_m : S_m \in \mathcal{X}^m\}$ such that*

$$\mathcal{E}_m(S_m) = \phi_m(\widehat{P}_m) \qquad \textit{for all } S_m \in \mathcal{X}^m.$$

Proof. Let $\widehat{P}_m : \mathcal{X}^m \to \mathcal{P}(\mathcal{X})$ be the empirical-measure map and equip its image $\{\widehat{P}_m(S_m) : S_m \in \mathcal{X}^m\}$ with the quotient σ-algebra induced by $\widehat{P}_m$. If $\widehat{P}_m(S_m) = \widehat{P}_m(S'_m)$, then S'_m is a permutation of S_m (as a multiset with multiplicity), so permutation invariance implies $\mathcal{E}_m(S_m) = \mathcal{E}_m(S'_m)$; hence $\mathcal{E}_m$ is constant on fibers of $\widehat{P}_m$. Therefore there exists a well-defined measurable ϕ_m on the image of $\widehat{P}_m$ such that $\mathcal{E}_m = \phi_m \circ \widehat{P}_m$, i.e. $\mathcal{E}_m(S_m) = \phi_m(\widehat{P}_m)$. □

Proposition A.3 (Permutation + proportional invariance ⇒ a single functional)**.** *Let $(\mathcal{E}_m)_{m\geq 1}$ be permutation and proportionally invariant. Then there exists a single measurable map ϕ defined on the set of all empirical measures $\bigcup_{m\geq 1}\{\widehat{P}_m : S_m \in \mathcal{X}^m\}$ such that*

$$\mathcal{E}_m(S_m) = \phi(\widehat{P}_m) \qquad \textit{for all } m \geq 1,\ S_m \in \mathcal{X}^m.$$

Proof. By Prop. A.2, for each m there exists ϕ_m with $\mathcal{E}_m(S_m) = \phi_m(\widehat{P}_m)$. Define ϕ on empirical measures by $\phi(\widehat{P}_m) := \phi_m(\widehat{P}_m)$. If the same empirical measure ν can be represented at two sample sizes m and m', then necessarily $m' = km$ for some integer k and the m'-sample is obtained by duplicating each of the m atoms k times. Proportional invariance then implies $\phi_m(\nu) = \phi_{m'}(\nu)$, so ϕ is well defined. □

A.1.2. Encoder CLT via the functional delta method

Let $P \in \mathcal{P}(\mathcal{X})$ and let $X_1, X_2, \ldots \overset{\text{iid}}{\sim} P$ with empirical measure $P_m = \frac{1}{m}\sum_{i=1}^{m} \delta_{X_i}$. Assume the encoder is distributionally invariant, so that $\mathcal{E}(S_m) = \phi(P_m)$ for a functional $\phi : \mathcal{P}(\mathcal{X}) \to \mathbb{R}^d$.

Theorem A.4 (Encoder CLT). *Suppose $\phi : \mathcal{P}(\mathcal{X}) \to \mathbb{R}^d$ is Hadamard differentiable at P tangentially to a subspace containing the empirical process limit, and denote its derivative by $D\phi_P$. Then*

$$\sqrt{m}\left(\phi(P_m) - \phi(P)\right) \overset{d}{\Rightarrow} D\phi_P(\mathbb{G}_P),$$

where $\mathbb{G}_P$ is the P-Brownian bridge. In particular, if $D\phi_P(\mathbb{G}_P)$ is a mean-zero Gaussian in $\mathbb{R}^d$ with covariance Σ_P, then

$$\sqrt{m}\left(\mathcal{E}(S_m) - \phi(P)\right) \overset{d}{\Rightarrow} \mathcal{N}(0, \Sigma_P),$$

which is Eqn. 1.

Proof. By the classical empirical process CLT, $\sqrt{m}\,(P_m - P) \Rightarrow \mathbb{G}_P$ in an appropriate function space (e.g. $\ell^\infty(\mathcal{F})$ for a Donsker class $\mathcal{F}$), see Van Der Vaart & Wellner (1996, Ch. 2). Hadamard differentiability of ϕ at P allows application of the functional delta method Van Der Vaart & Wellner (1996, Thm. 3.9.4), yielding the stated limit. See Fishman et al. (2025) for this argument in the GDE setting. □

A.1.3. Plug-in losses for conditioned models

We now formalize the plug-in CLTs used to justify minibatch training in Alg. 1 and Prop. 3.1. The core plug-in loss CLT and bias expansion are proved in Fishman et al. (2025, Assump. (C.1)–(C.3) and Thm. "Large-m behaviour of the plug-in loss"). Here we record a DCT-specialized statement and explicitly handle two technical points needed for conditional transport: (i) joint CLTs for *pairs* of embeddings when the source and target minibatches are either independent (product coupling) or *paired/coupled* (non-product coupling), and (ii) the fact that reusing a *finite* number of points from the same minibatch as the embedding does not affect the limiting distribution.

Theorem A.5 (Plug-in loss CLTs for DCT). *Fix indices (u, v). Let $z_u^\star := \phi(P_u)$ and $z_v^\star := \phi(P_v)$. Consider either of the following sampling schemes:*

*(A) (**Independent minibatches**) $\hat{S}_u = (X_{u1}, \dots, X_{um}) \sim P_u^{\otimes m}$ and $\hat{S}_v = (X_{v1}, \dots, X_{vm'}) \sim P_v^{\otimes m'}$ are independent.*

*(B) (**Coupled/paired minibatches**) $m' = m$ and $(X_{ui}, X_{vi})_{i=1}^m \overset{\text{iid}}{\sim} \Pi_{uv}$ for some coupling Π_{uv} with marginals P_u and P_v, with $\hat{S}_u = (X_{u1}, \dots, X_{um})$ and $\hat{S}_v = (X_{v1}, \dots, X_{vm})$.*

Define $z_u := \mathcal{E}(\hat{S}_u)$ and $z_v := \mathcal{E}(\hat{S}_v)$. When $m' = m$, case (A) is obtained from case (B) by taking the product coupling $\Pi_{uv} = P_u \otimes P_v$.

Assume the encoder is asymptotically linear at P_u and P_v as in Fishman et al. (2025, Assump. (C.1)): there exist $\psi_u \in L^2(P_u)$ and $\psi_v \in L^2(P_v)$ with mean zero such that

$$\sqrt{m}\,(z_u - z_u^\star) = \frac{1}{\sqrt{m}} \sum_{i=1}^{m} \psi_u(X_{ui}) + o_{\mathbb{P}}(1), \qquad \sqrt{m'}\,(z_v - z_v^\star) = \frac{1}{\sqrt{m'}} \sum_{i=1}^{m'} \psi_v(X_{vi}) + o_{\mathbb{P}}(1).$$

Let $\Sigma_u := \mathrm{Var}_{X \sim P_u}[\psi_u(X)]$ and $\Sigma_v := \mathrm{Var}_{X \sim P_v}[\psi_v(X)]$. In case (B), define the cross-covariance $\Sigma_{uv}^{12} := \mathrm{Cov}_{(X,Y) \sim \Pi_{uv}}[\psi_u(X), \psi_v(Y)]$.

Fix $x \in \mathcal{X}$ and let $\ell^{\mathrm{sc}}(x, \cdot) : \mathbb{R}^d \to \mathbb{R}$ and $\ell^{\mathrm{stc}}(x, \cdot, \cdot) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ be differentiable at $z_u^\star$ and $(z_u^\star, z_v^\star)$, respectively. Define the plug-in losses

$$\widehat{\ell}_m^{\mathrm{sc}}(x) := \ell^{\mathrm{sc}}(x, z_u), \qquad \widehat{\ell}_{m,m'}^{\mathrm{stc}}(x) := \ell^{\mathrm{stc}}(x, z_u, z_v).$$

Then:

*(i) **Joint embedding CLT.** $\left(\sqrt{m}(z_u - z_u^\star), \sqrt{m'}(z_v - z_v^\star)\right) \Rightarrow \mathcal{N}(0, \Sigma_{uv})$. In case (A), $\Sigma_{uv} = \mathrm{diag}(\Sigma_u, \Sigma_v)$, while in case (B) (with $m' = m$) we have*

$$\Sigma_{uv} = \begin{pmatrix} \Sigma_u & \Sigma_{uv}^{12} \\ (\Sigma_{uv}^{12})^\top & \Sigma_v \end{pmatrix}.$$

*(ii) **Source-conditioned plug-in loss.** $\sqrt{m}\big(\widehat{\ell}_m^{\mathrm{sc}}(x) - \ell^{\mathrm{sc}}(x, z_u^\star)\big) \Rightarrow \mathcal{N}(0, \sigma_{\mathrm{sc}}^2(x))$.*

(iii) ***Source–target conditioned plug-in loss.*** *If* $m/m' \to \rho \in (0,\infty)$ *then* $\sqrt{m}\big(\widehat{\ell}^{\mathrm{stc}}_{m,m'}(x) - \ell^{\mathrm{stc}}(x, z_u^\star, z_v^\star)\big) \Rightarrow \mathcal{N}(0, \sigma^2_{\mathrm{stc},\rho}(x))$.

Moreover, the same limits in (ii)–(iii) continue to hold if x *is replaced by any fixed finite subset of points from* $\hat{S}_u$ *(e.g.* $x = X_{u1}$*): conditioning on those points changes* $\sqrt{m}(z_u - z_u^\star)$ *only by* $o_{\mathbb{P}}(1)$.

Proof. (i) The asymptotic linearity expansions reduce the joint fluctuations to sums of i.i.d. terms. In case (A) the two sums are independent, giving the block-diagonal covariance. In case (B), apply the multivariate CLT to the i.i.d. sequence $(\psi_u(X_{ui}), \psi_v(X_{vi}))_{i=1}^m$ to obtain a joint Gaussian limit with covariance as stated.

(ii)–(iii) Apply the (multivariate) delta method to the map $z \mapsto \ell^{\mathrm{sc}}(x, z)$ in the source-conditioned case and to $(z, z') \mapsto \ell^{\mathrm{stc}}(x, z, z')$ in the source–target-conditioned case, using the joint CLT from (i). For (iii), note that $\sqrt{m}(z_v - z_v^\star) = \sqrt{m/m'} \cdot \sqrt{m'}(z_v - z_v^\star) \Rightarrow \sqrt{\rho}\, Z_v$.

(reuse-of-points dependence) The expansion in Assump. (C.1) already gives $\sqrt{m}(z_u - z_u^\star) = m^{-1/2} \sum_{i=1}^m \psi_u(X_{ui}) + o_{\mathbb{P}}(1)$. If we condition on X_{u1} (or any fixed finite subset), only finitely many summands are affected and they vanish at rate $m^{-1/2}$, so the conditional limit of $\sqrt{m}(z_u - z_u^\star)$ is unchanged. Applying the same delta-method argument conditional on X_{u1} yields the stated conclusion. □

Corollary A.6 (Mean consistency and (asymptotic) unbiasedness of plug-in training)**.** *Under the assumptions of Thm. A.5, suppose additionally that for each fixed* x *the maps* $z \mapsto \ell^{\mathrm{sc}}(x, z)$ *and* $(z, z') \mapsto \ell^{\mathrm{stc}}(x, z, z')$ *are locally Lipschitz near* $z_u^\star$ *and* $(z_u^\star, z_v^\star)$*, respectively, and that* $\sup_m \mathbb{E}\|\sqrt{m}(z_u - z_u^\star)\|^2 < \infty$ *and* $\sup_{m'} \mathbb{E}\|\sqrt{m'}(z_v - z_v^\star)\|^2 < \infty$*. Then* $\mathbb{E}\big|\widehat{\ell}^{\mathrm{sc}}_m(x) - \ell^{\mathrm{sc}}(x, z_u^\star)\big| \to 0$ *and* $\mathbb{E}\big|\widehat{\ell}^{\mathrm{stc}}_{m,m'}(x) - \ell^{\mathrm{stc}}(x, z_u^\star, z_v^\star)\big| \to 0$*, hence the plug-in objectives are mean consistent. If, moreover, these maps are twice continuously differentiable with bounded Hessian near the same points and* $\mathbb{E}[z_u - z_u^\star] = O(m^{-1})$ *and* $\mathbb{E}[z_v - z_v^\star] = O(m'^{-1})$*, then the plug-in biases are* $O(m^{-1})$ *(and* $O(m'^{-1})$*).*

Proof. This is the same local Lipschitz / second-order Taylor expansion argument as in Fishman et al. (2025, Thm. "Large-m behaviour of the plug-in loss"), applied to the scalar maps $z \mapsto \ell^{\mathrm{sc}}(x, z)$ and $(z, z') \mapsto \ell^{\mathrm{stc}}(x, z, z')$ together with the moment bounds implied by the encoder CLTs. □

A.2. When do source samples matter?

The plug-in CLTs above justify minibatch training for objectives that compare *distributions*. Separately, when training any-to-any models with an *independent* pairing policy (product coupling) and a stochastic generator that has access to additional noise, the population objective can admit solutions that match the target distribution while effectively ignoring the source sample. This subsection records a simple identifiability observation and a practical diagnostic.

Proposition A.7 (Degenerate conditional sampler under product coupling)**.** *Fix indices* (u, v)*. Let* $\xi \sim \nu$ *be auxiliary noise independent of* $X \sim P_u$*. Consider a source–target conditioned generator* $\mathcal{T}(\cdot \mid z_u, z_v; \xi)$ *and a distribution-level loss of the form*

$$\mathcal{L}_{\mathrm{stc}}(u, v) := \mathcal{L}\Big(P_v,\ \mathsf{Law}\big(\mathcal{T}(X \mid z_u, z_v; \xi)\big)\Big),$$

where $\mathcal{L}(P, Q) = 0$ *iff* $P = Q$*. If the model class contains* x-ignoring *generators of the form*

$$\mathcal{T}(x \mid z_u, z_v; \xi) = G(z_v; \xi)$$

and for each v *there exists such a* G *with* $G(z_v; \xi) \sim P_v$*, then the minimum of* $\mathcal{L}_{\mathrm{stc}}(u, v)$ *equals* 0 *and is achieved by an* x*-ignoring solution.*

Proof. If $\mathcal{T}(x \mid z_u, z_v; \xi) = G(z_v; \xi)$ and $G(z_v; \xi) \sim P_v$, then the output marginal distribution equals P_v regardless of P_u. Thus $\mathcal{L}(P_v, \mathsf{Law}(\mathcal{T}(X \mid z_u, z_v; \xi))) = 0$. □

Prop. A.7 does *not* say a model must ignore x; it says a product-coupling, distribution-level objective cannot generally rule out this failure mode when the generator has sufficient stochasticity. In practice, one must break this symmetry—e.g. by restricting to deterministic generators; using structured pairings/couplings (e.g. optimal transport) (Tong et al., 2023b); or adding regularization that selects a coupling or geometry (cf. (Haviv et al., 2024)).

Generator	d_{pair}	d_{rand}	$d_{\text{pair}}/d_{\text{rand}}$	Rank ρ
Flow matching (FM)	0.011	0.629	0.017	1.000
Energy regression (Energy)	0.543	0.615	0.885	0.106
Sliced Wasserstein (SWD)	0.555	0.623	0.893	0.108
Stochastic energy sampler	0.647	0.649	0.998	0.009

Table 5. Source-sample alignment diagnostic on the MVN any-to-any benchmark (App. E, K=1000 training distributions; averages over 20 random (u, v) pairs with N=200 samples each; d_{rand} averaged over 50 random permutations). FM (which uses a sample-level paired objective) yields strong alignment ($d_{\text{pair}}/d_{\text{rand}} \approx 0.02$, $\rho \approx 1$). Energy and SWD are deterministic maps at inference but are trained with set-level distributional losses; in this experiment they show moderate alignment ($d_{\text{pair}}/d_{\text{rand}} \approx 0.9$, $\rho \approx 0.1$). The stochastic energy sampler injects noise at inference and achieves $d_{\text{pair}} \approx d_{\text{rand}}$, consistent with the x-ignoring failure mode in Prop. A.7 and Lem. A.8.

An alignment diagnostic. Fix a cost $c : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_+$ (we use $c(x, y) = \|x - y\|_2$). For a fixed pair (u, v), let $X \sim P_u$ and $Y \sim P_v$ be independent and let $\hat{Y} := \mathcal{T}(X \mid z_u, z_v; \xi)$ for independent noise $\xi \sim \nu$. Let $\mu_u := \mathbb{E}[X]$ and $\mu_v := \mathbb{E}[Y]$ and define the centered cost

$$\bar{c}_{uv}(x, y) := c(x - \mu_u,\ y - \mu_v).$$

We report

$$d_{\text{pair}} := \mathbb{E}\big[\bar{c}_{uv}(X, \hat{Y})\big], \qquad d_{\text{rand}} := \mathbb{E}\big[\bar{c}_{uv}(X, Y)\big], \qquad \text{and the ratio } d_{\text{pair}}/d_{\text{rand}}.$$

Centering removes the bulk mean-shift component (which can otherwise dominate c) and makes the diagnostic more sensitive to the coupling.

Lemma A.8 (Independence baseline)**.** *If $\hat{Y}$ is independent of X and $\mathsf{Law}(\hat{Y}) = P_v$, then $d_{\text{pair}} = d_{\text{rand}}$.*

Proof. Under the assumptions, $(X, \hat{Y}) \overset{d}{=} (X, Y)$ with $Y \sim P_v$ independent of X, so the expectations agree. □

As a complementary diagnostic, let $\Delta_{uv} := \mu_v - \mu_u$ and compute the Spearman rank correlation ρ between the one-dimensional projections $\langle X, \Delta_{uv}\rangle$ and $\langle \hat{Y}, \Delta_{uv}\rangle$. High $|\rho|$ indicates that the map preserves rank structure along the mean-shift direction, whereas $\rho \approx 0$ is consistent with weak pointwise dependence. In practice we estimate the expectations and means with sample averages. For the *stochastic energy sampler* baseline in Table 5, we augment the deterministic Energy regression map with per-sample noise (so $\hat{Y} = \mathcal{T}(X \mid z_u, z_v; \xi)$ with $\xi \sim \nu$), making the model class expressive enough to realize x-ignoring solutions.

Taken together, these results illustrate when we can expect DCT to learn more than an independent conditional sampler: objectives with an explicit sample-level pairing signal (or additional structure selecting a coupling) can induce meaningful pointwise dependence, while purely distributional objectives need not.

A.3. How is the latent space structured?

Following the manifold/trajectory viewpoint of Fishman et al. (2025), we use the learned embedding space as a coordinate system for a family of related distributions and probe it via *latent interpolants*. Fix a pair (u, v) with embeddings (z_u, z_v). We linearly interpolate between them, $z(t) = (1 - t)z_u + tz_v$, choose a grid $0 = t_0 < \cdots < t_K = 1$, and set $z_k = z(t_k)$. Starting from samples $x_0 \sim P_u$, we form a discrete trajectory by repeatedly applying the source–target conditioned map between successive latent codes, i.e. $x_{k+1} = \mathcal{T}(x_k \mid z_k, z_{k+1})$. The empirical laws of $(x_k)_{k=0}^K$ define a path in distribution space.

Fig. 6 shows the resulting STC trajectory on the MVN any-to-any benchmark (App. E) alongside the closed-form optimal transport (OT) displacement interpolation between the same endpoint Gaussians. In this example, the learned trajectory closely tracks the OT path, suggesting that linear structure in the latent space induces a meaningful geometry in distribution space (at least within this controlled family). This is not a guarantee, it merely shows that the latent spaces learned by the distribution encoders for the any-to-any transport task inherit many of the interesting and useful properties of GDEs.

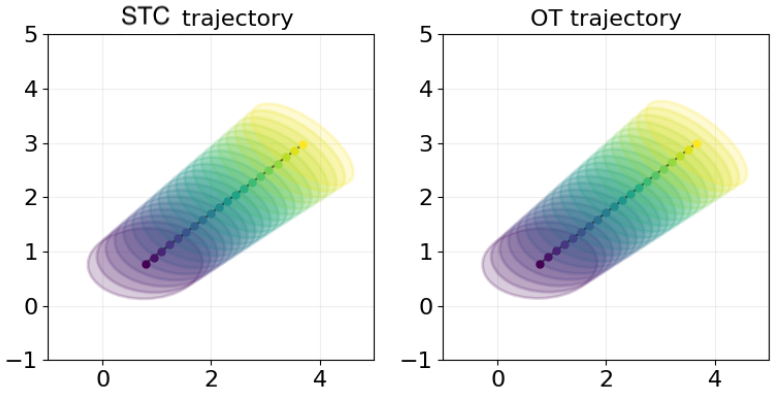

Figure 6. Latent-space interpolants on the MVN any-to-any benchmark. **Left:** STC trajectory obtained by sequentially pushing samples using intermediate latent codes along a linear interpolant between endpoint embeddings. **Right:** OT displacement interpolation (computed in closed form for Gaussians).

B. Basline Comparisons

B.1. Unsupervised Transport

In our unsupervised transport experiments, we compare source-target-condition models in the DCT framework to K-to-K baselines, which condition on one-hot embeddings of distributions. This approach is closely related to that of MMSI (Albergo et al., 2023a), but distinct in a subtle way.

MMSI models interpolant time on the simplex, where each point in the simplex corresponds to an interpolant taking a convex combination of a sample from each of the K distributions proportional to their simplicial weight. Our K-to-K baselines (when using flow matching generators) can be viewed as MMSI models which take "edge paths" on the simplex, where we flow directly between two corners without any weight on other distributions.

We choose our one-hot K-to-K as a baseline, because it allows for direct comparison across a broader family of transport mechanisms rather than flow matching alone, and anecdotally we have observed more stable performance with MMSI models restricted to simplicial edge paths.

B.2. Supervised Transport

In our supervised transport experiments, we evaluate source-conditioned models. These models are closely related to Meta Flow Matching models (Atanackovic et al., 2024). When using flow matching, source-conditioned DCTs differ from MFM only in the choice of distribution encoder. Rather than a distributionally invariant encoder, MFM instead uses a heuristic "population encoder" which involves message passing over a k-nearest-neighbor graph on the input population.

We choose to baseline against source-conditioned models with distributionally invariant encoders as these provide a CLT and robustness to variation in sample set size. Furthermore, using source-conditioned models rather than MFM alone allows us to again directly compare between source-conditioned and source-target-conditioned models across a broad range of transport mechanisms.

C. Standard distribution-conditioned transport maps

Several experiments in this paper reuse the same generator families and only vary the dataset, encoder, or model capacity. This appendix defines the three *continuous-data* generators used throughout: sliced Wasserstein regression (**SWD**) (Kolouri et al., 2019), energy/MMD regression (**Energy**) (Gneiting & Raftery, 2007; Sutherland et al., 2016), and flow matching (**FM**) (Liu et al., 2022; Lipman et al., 2023; Albergo et al., 2023b). Discrete sequence generators for the TCR experiments are described separately in Appendix J.3. Where an experiment uses different hyperparameters (e.g. number of projections or solver tolerances), it is stated explicitly in that experiment's appendix.

Terminology. Throughout, **DCT** refers to the overall *distribution-conditioned transport* framework of coupling a distribution encoder with a conditional generator. Within DCT we use **source-conditioned** (**SC**) to mean generators conditioned only on the source embedding z_{src}, and **source–target-conditioned** (**STC**) to mean generators conditioned on both $(z_{\text{src}}, z_{\text{tgt}})$; STC models are also referred to as *any-to-any* models in the main text. Unless otherwise stated, the generator families below apply to both SC and STC, with SC obtained by dropping z_{tgt}.

C.1. Notation

Let $S^{\text{src}} = \{x_j\}_{j=1}^n \sim P^{\text{src}}$ and $S^{\text{tgt}} = \{y_j\}_{j=1}^n \sim P^{\text{tgt}}$ denote source and target sample sets (not necessarily paired). Let $z_{\text{src}} = \mathcal{E}(S^{\text{src}})$ and $z_{\text{tgt}} = \mathcal{E}(S^{\text{tgt}})$ be the corresponding distribution embeddings. All generators produce a transported sample set $\widehat{S}^{\text{tgt}} = \{\hat{y}_j\}_{j=1}^n$; in the STC setting they are conditioned on $(z_{\text{src}}, z_{\text{tgt}})$, while in the SC setting they are conditioned on z_{src} alone.

C.2. Standard distribution encoder

Across continuous-data experiments, we use a single *standard* distribution encoder $\mathcal{E}$ to map a finite sample set to a fixed-dimensional embedding. Concretely, given a set $S = \{x_j\}_{j=1}^n \subset \mathbb{R}^d$, the encoder outputs $z = \mathcal{E}(S) \in \mathbb{R}^{d_z}$. We adopt the mean-pooled DeepSets/GNN-style architecture used in Generative Distribution Embeddings (GDE) (Fishman et al., 2025), which is consistently strong across their benchmark suite.

Architecture. We first compute per-sample features with an MLP,

$$h_j^{(0)} = \text{MLP}_{\text{in}}(x_j) \in \mathbb{R}^{d_h}. \tag{6}$$

We then apply L mean-pooled update blocks. For $\ell = 1, \ldots, L$, define the pooled representation $\bar{h}^{(\ell-1)} = \frac{1}{n}\sum_{j=1}^n h_j^{(\ell-1)}$ and update each sample feature by concatenating with the pooled representation:

$$h_j^{(\ell)} = \text{MLP}_{\text{pool}}^{(\ell)}\left(\left[h_j^{(\ell-1)}; \bar{h}^{(\ell-1)}\right]\right). \tag{7}$$

Finally, we average over samples and project to the latent space:

$$z = \sigma\left(W \cdot \frac{1}{n}\sum_{j=1}^n h_j^{(L)} + b\right), \tag{8}$$

with $\sigma = \text{SELU}$. Optionally, we ℓ_2-normalize z; this is used only when stated explicitly.

Hyperparameters. Unless stated otherwise in an experiment appendix, we use $L = 2$ mean-pooled blocks and 2-layer MLPs with SELU activations. The hidden dimension d_h and embedding dimension d_z are reported per experiment.

Non-continuous data. For discrete sequence experiments (TCR forecasting), we use an ESM2-based distribution encoder described in Appendix J.2.

C.3. Deterministic regression generators (SWD and Energy)

Both SWD and Energy use a deterministic transport map applied pointwise:

$$\hat{y}_j = \mathcal{T}_\theta(x_j \mid z_{\text{src}}, z_{\text{tgt}}). \tag{9}$$

They differ only in the training objective, which compares $\widehat{S}^{\text{tgt}}$ to the observed target set S^{tgt} using a distributional distance (Appendix D).

SWD (sliced Wasserstein regression). We minimize the empirical sliced 2-Wasserstein distance using $L = 100$ random projections $\{\omega_\ell\}_{\ell=1}^L$ drawn uniformly from the unit sphere. For each ω_ℓ, we compute scalar projections and sort them, $\text{sort}(\{\langle \omega_\ell, \hat{y}_j\rangle\}_{j=1}^n)$ and $\text{sort}(\{\langle \omega_\ell, y_j\rangle\}_{j=1}^n)$, then take the mean squared difference across matched order statistics and projections.

Energy (energy/MMD regression). We minimize the energy distance, equivalently MMD with the energy kernel $k(x, y) = -\|x - y\|$ (Appendix D). The energy kernel is parameter-free, avoiding bandwidth selection.

C.4. Flow matching generator (FM)

Flow matching parameterizes transport with a time-conditional velocity field $v_\theta(x, t, z_{\text{src}}, z_{\text{tgt}})$. For training we sample $t \sim \text{Uniform}[0, 1]$, $x_0 \sim P^{\text{src}}$, $x_1 \sim P^{\text{tgt}}$, and $\epsilon \sim \mathcal{N}(0, I)$, and form the noisy linear interpolation

$$x_t = (1 - t)x_0 + tx_1 + \sigma\epsilon, \tag{10}$$

with $\sigma = 0.5$ unless otherwise stated. We train with the conditional flow matching objective

$$\mathcal{L}_{\text{FM}} = \mathbb{E}\Big[\big\|v_\theta(x_t, t, z_{\text{src}}, z_{\text{tgt}}) - (x_1 - x_0)\big\|_2^2\Big]. \tag{11}$$

At inference, for each source sample $x(0) = x_0$ we solve the ODE

$$\frac{dx}{dt} = v_\theta(x, t, z_{\text{src}}, z_{\text{tgt}}), \qquad t \in [0, 1], \tag{12}$$

and take $\hat{y} = x(1)$. We use an adaptive Dormand–Prince solver (dopri5) with absolute and relative tolerances 10^{-4} unless otherwise stated.

D. Standard evaluation metrics

Across experiments we report three distributional distances between a generated sample set $\widehat{S} = \{\hat{y}_i\}_{i=1}^n$ and a ground-truth target set $S = \{y_j\}_{j=1}^n$: sliced Wasserstein distance (**SWD**), energy distance (**Energy**), and MMD with an RBF kernel (**MMD-RBF**). These metrics are computed in the data space unless stated otherwise (e.g. in the TCR experiments, they are computed in ESM2 embedding space).

D.1. Energy distance (MMD with energy kernel)

The energy distance between distributions P and Q can be written as

$$D_{\text{energy}}(P, Q) = 2\mathbb{E}\|X - Y\| - \mathbb{E}\|X - X'\| - \mathbb{E}\|Y - Y'\|, \tag{13}$$

where $X, X' \sim P$ i.i.d. and $Y, Y' \sim Q$ i.i.d. We estimate this quantity empirically from the two finite sample sets. This is equivalent to MMD with kernel $k(x, y) = -\|x - y\|$. The energy distance is closely related to the *energy score*, a strictly proper scoring rule for probabilistic prediction (Gneiting & Raftery, 2007).

D.2. Sliced 2-Wasserstein distance (SWD)

We use the standard sliced 2-Wasserstein distance (Kolouri et al., 2019) estimated with $L = 100$ random projections $\{\omega_\ell\}_{\ell=1}^L$ drawn uniformly from the unit sphere. For each projection, we compute and sort the scalar projections of both sets, match order statistics, and average the squared differences across projections:

$$\widehat{\text{SWD}}_2^2(\widehat{S}, S) = \frac{1}{L}\sum_{\ell=1}^{L}\frac{1}{n}\sum_{j=1}^{n}\left(\text{sort}(\{\langle\omega_\ell, \hat{y}_i\rangle\}_{i=1}^n)_j - \text{sort}(\{\langle\omega_\ell, y_i\rangle\}_{i=1}^n)_j\right)^2. \tag{14}$$

We report $\widehat{\text{SWD}}_2(\widehat{S}, S) = \sqrt{\widehat{\text{SWD}}_2^2(\widehat{S}, S)}$.

D.3. MMD with RBF kernel (MMD-RBF)

Given a positive definite kernel k, the squared MMD is

$$\text{MMD}^2(P, Q) = \mathbb{E}[k(X, X')] + \mathbb{E}[k(Y, Y')] - 2\mathbb{E}[k(X, Y)], \tag{15}$$

with $X, X' \sim P$ and $Y, Y' \sim Q$ i.i.d., estimated empirically from samples. MMD is a standard kernel two-sample statistic (Gretton et al., 2012). We use an RBF kernel

$$k_{\text{RBF}}(x, y) = \exp\Big(-\tfrac{\|x-y\|_2^2}{2\sigma^2}\Big), \tag{16}$$

with bandwidth σ set by the median heuristic on the pooled samples from $\widehat{S} \cup S$.

E. Gaussian and Gaussian Mixture Experiments

This appendix provides experimental details for the Gaussian experiments in Sections 5.1 and 6.1. We describe the data-generating processes, model architectures, training configurations, and evaluation protocols. Generator families and metrics are defined in Appendices C and D.

E.1. Data-Generating Process

E.1.1. Multivariate Normal (MVN) Distributions

We construct a dataset of $n = 50{,}000$ sample sets, each drawn from a distinct bivariate normal distribution. The parameters for each distribution i are sampled as follows:

Mean Prior. Each component of the mean vector is drawn independently from a uniform distribution:

$$\mu_i \sim \mathrm{Uniform}([0, 5]^2) \tag{17}$$

Covariance Prior. The covariance matrix is sampled from an inverse-Wishart distribution:

$$\Sigma_i \sim \mathrm{InverseWishart}(\nu, \Psi) \tag{18}$$

where the degrees of freedom $\nu = 10 \cdot (d - 1) = 10$ for $d = 2$ dimensions, and the scale matrix $\Psi = I_d$ (identity matrix). This parameterization ensures well-conditioned covariance matrices with moderate variability.

Sample Generation. For each distribution i, we draw $n_{\text{set}} = 100$ i.i.d. samples:

$$x_{ij} \sim \mathcal{N}(\mu_i, \Sigma_i), \quad j = 1, \ldots, n_{\text{set}} \tag{19}$$

Sampling is performed via the Cholesky decomposition $\Sigma_i = L_i L_i^\top$:

$$x_{ij} = \mu_i + L_i z_{ij}, \quad z_{ij} \sim \mathcal{N}(0, I_d) \tag{20}$$

Unique Distribution Subsets. For experiments studying generalization across distribution density (varying K), we sample $K \in \{10, 100, 1{,}000, 10{,}000\}$ unique parameter sets (μ_k, Σ_k) and repeat each parameter set n/K times. This creates multiple sample sets from the same underlying distribution, enabling the study of how models perform when training distributions are seen multiple times versus once.

E.1.2. Gaussian Mixture Models (GMM)

We extend the MVN setting to Gaussian mixture models with $C = 3$ components. The parameters for each GMM i are sampled as follows:

Mixture Weight Prior. Component weights are drawn from a symmetric Dirichlet distribution:

$$\pi_i \sim \mathrm{Dirichlet}(\alpha \cdot \mathbf{1}_C) \tag{21}$$

where $\alpha = 1.0$ yields a uniform prior over the simplex.

Component Mean Prior. Each component mean is drawn independently:

$$\mu_{ic} \sim \mathrm{Uniform}([0, 5]^2), \quad c = 1, \ldots, C \tag{22}$$

Component Covariance Prior. Each component covariance is drawn independently:

$$\Sigma_{ic} \sim \mathrm{InverseWishart}(\nu, \Psi), \quad c = 1, \ldots, C \tag{23}$$

with the same hyperparameters as the MVN case ($\nu = 10$, $\Psi = I_d$).

Sample Generation. For each GMM i, we draw $n_{\text{set}} = 1{,}000$ samples via ancestral sampling:

$$c_{ij} \sim \mathrm{Categorical}(\pi_i) \tag{24}$$

$$x_{ij} \sim \mathcal{N}(\mu_{i,c_{ij}}, \Sigma_{i,c_{ij}}) \tag{25}$$

The larger set size (1,000 vs. 100 for MVN) provides sufficient samples to capture the multimodal structure.

E.2. Supervised Gaussian Experiments

For the semi-supervised experiments, we construct paired source-target datasets with known transformations.

E.2.1. Supervised MVN Dataset

The supervised MVN dataset creates paired distributions via a fixed shift transformation:

$$Y = X + b, \quad b = (1, 1)^\top \tag{26}$$

Source Distribution. To study the generalization beyond the training distribution and evaluate the value of any-to-any models for generalization, source means are restricted:

$$\mu_i^{\text{src}} \sim \text{Uniform}([0, 2.5]^2) \tag{27}$$

Target Generation. Given source samples $\{x_{ij}\}$, target samples are:

$$y_{ij} = x_{ij} + b \tag{28}$$

The target samples are then randomly permuted to remove point-wise correspondence, creating an unpaired but distributionally related dataset.

E.2.2. Supervised GMM Dataset

The supervised GMM dataset creates a more challenging bimodal target structure:

$$Y = X + b \pm b_{\text{off}} \tag{29}$$

where $b = (1, 1)^\top$ and $b_{\text{off}} = 0.1 \cdot (-1, 1)^\top$.

Target Generation. For each source sample, we create two shifted versions:

$$y_{ij}^+ = x_{ij} + b + b_{\text{off}} \tag{30}$$

$$y_{ij}^- = x_{ij} + b - b_{\text{off}} \tag{31}$$

These are concatenated and subsampled to maintain the original set size, then randomly permuted. This creates a bimodal target distribution from a unimodal (per-component) source, testing the model's ability to learn complex distributional transformations.

E.3. Model Architecture

E.3.1. Distribution Encoder

We use the standard distribution encoder described in Appendix C.2 (mean-pooled DeepSets/GNN encoder). For MVN experiments, we use hidden dimension $d_h = 64$ and latent dimension $d_z = 16$; for GMM, we increase these to $d_h = 256$ and $d_z = 128$ to accommodate the more complex multimodal structure.

E.3.2. Embedding Encoder (Baseline)

For the K-to-K baseline that treats each distribution as a discrete label, we use a learnable embedding table:

$$z_i = \text{Embedding}(i) \in \mathbb{R}^{d_z} \tag{32}$$

where $i \in \{1, \ldots, K\}$ is the distribution index. This encoder cannot generalize to unseen distributions at test time.

E.3.3. Transport Map

The transport map $\mathcal{T}(x \mid z_{\text{src}}, z_{\text{tgt}})$ is a 4-layer MLP with SELU activations that takes the concatenation of a source sample x with both the source and target distribution embeddings: $\hat{y} = \text{MLP}([x; z_{\text{src}}; z_{\text{tgt}}])$. The hidden dimension matches the encoder ($d_h = 64$ for MVN, $d_h = 256$ for GMM).

We compare the three standard continuous-data generator families defined in Appendix C: sliced Wasserstein regression (SWD), energy/MMD regression (Energy), and flow matching (FM).

E.4. Training Protocol

All models are trained using Adam with a fixed learning rate of 2×10^{-4} for 200 epochs. We use a batch size of 256 distribution pairs, where each pair consists of source and target sample sets. No learning rate scheduling or early stopping is applied.

Pair Sampling. During training, we sample source-target pairs uniformly at random. For the unsupervised any-to-any setting, source and target distributions are sampled independently from the full dataset, so the model sees arbitrary pairings. For the supervised setting, each target is deterministically derived from its corresponding source via the fixed transformation (shift for MVN, shift plus off-axis displacement for GMM).

Bidirectional vs. Unidirectional Loss. The unsupervised any-to-any objective trains transport in both directions, summing the losses:

$$\mathcal{L}_{\text{unsup}} = \mathcal{L}(\text{src} \to \text{tgt}) + \mathcal{L}(\text{tgt} \to \text{src}) \tag{33}$$

This symmetric objective encourages the encoder to learn representations that support transport regardless of which distribution serves as source or target. For supervised training, we use only the forward direction $\mathcal{L}_{\text{sup}} = \mathcal{L}(\text{src} \to \text{tgt})$, since the transformation is inherently asymmetric.

E.5. Evaluation Protocol

E.5.1. Unsupervised Evaluation

We evaluate transport quality in two settings. For *in-distribution* evaluation, we test on held-out source-target pairs where both distributions appeared during training, though not necessarily paired together. For *out-of-distribution* evaluation, we fix a source distribution from the training set and transport to a dense grid of novel target distributions with means $\mu \in [0, 5]^2$ and randomly sampled covariances. This tests zero-shot generalization to target distributions never seen during training.

We measure transport quality using the energy distance (Appendix D), estimated empirically from the finite sample sets.

E.5.2. Semi-Supervised Evaluation

For the semi-supervised experiments, we restrict the supervised training data to distributions with source means satisfying $\|\mu\|_\infty \leq 2.5$, then evaluate across the full range $\|\mu\|_\infty \leq 5$. We partition test results by the L^∞ norm of the source mean to assess how performance degrades outside the supervised training support.

The semi-supervised approach first trains an any-to-any encoder on the full unsupervised dataset (all 50,000 distributions), then fits a ridge regression predictor $\hat{z}_{\text{tgt}} = W z_{\text{src}} + b$ on the paired subset within the restricted support. At test time, we encode the source distribution, predict the target embedding via the linear map, and apply the learned transport.

We compare against two baselines: a *supervised* model that conditions only on the source embedding and is trained exclusively on paired data within the support, and an *oracle* that uses the true target embedding computed by the encoder (providing an upper bound on semi-supervised performance).

E.6. Hyperparameter Summary

We summarize the key hyperparameters for reproducibility. All experiments use the same random seed protocol and data splits.

E.6.1. Dataset and Model Hyperparameters

Table 6 provides a consolidated view of the dataset and model configurations for both experimental settings.

For the supervised experiments, we restrict the mean prior to $[0, 2.5]^2$ so that the shifted targets remain within the original $[0, 5]^2$ range. The shift vector is $b = (1, 1)^\top$ for both MVN and GMM; GMM additionally uses an off-axis displacement of magnitude 0.1 to create bimodal targets.

Table 6. Dataset and model hyperparameters for Gaussian experiments.

Hyperparameter	**MVN**	**GMM**
Dataset		
Number of sample sets	50,000	50,000
Samples per set	100	1,000
Data dimension	2	2
Mean prior range	$[0, 5]^2$	$[0, 5]^2$
Covariance prior (inv-Wishart ν)	10	10
Mixture components	—	3
Dirichlet concentration α	—	1.0
Encoder		
Latent dimension d_z	16	128
Hidden dimension d_h	64	256
Mean-pooled GNN layers	2	2
FC layers per block	2	2
Transport Map		
MLP layers	4	4
Hidden dimension	64	256
Activation	SELU	SELU
Training		
Batch size	256	256
Learning rate	2×10^{-4}	2×10^{-4}
Epochs	200	200
Optimizer	Adam	Adam

E.7. Full Results

We present comprehensive results from our Gaussian experiments. All tables report energy distance (lower is better), averaged across three random seeds.

E.7.1. Unsupervised Transport: Scaling with K

Tables 7 and 8 report the full breakdown of transport quality as a function of K, the number of unique training distributions. The key observation is that the K-to-K baseline performs well on in-distribution targets when K is small (it can memorize each distribution), but degrades significantly on out-of-distribution targets. In contrast, the any-to-any encoder maintains consistent performance across both IID and OOD targets, demonstrating genuine distributional generalization.

E.7.2. Semi-Supervised Generalization

Tables 9 and 10 compare supervised, semi-supervised, and oracle methods on the paired Gaussian tasks. The supervised model is trained only on paired data within the restricted support ($\|\mu\|_\infty \leq 2.5$), while the semi-supervised approach uses the any-to-any encoder with a ridge predictor for the target embedding. The oracle uses the true target embedding.

The results show that the semi-supervised approach matches or exceeds supervised performance within the training support (IID) while maintaining substantially lower error outside it (OOD). This demonstrates that the any-to-any encoder learns distributional structure from the unsupervised objective that enables extrapolation even when the latent predictor must generalize beyond its training data.

Flow Matching Anomaly. We observe that supervised flow matching generalizes surprisingly well outside its training support in this synthetic setting (low OOD error even for the supervised method). This behavior is not replicated in real-world experiments, suggesting it may be an artifact of the simple linear transformation and low-dimensional setting where the learned flow field happens to extrapolate correctly.

Table 7. Distributional distances (↓) for MVN experiments across K unique distributions. We report MMD, SWD, and Energy distance.

IID Targets

	$K = 10$			$K = 100$			$K = 1,000$			$K = 10,000$		
Model	MMD	SWD	Energy	MMD	SWD	Energy	MMD	SWD	Energy	MMD	SWD	Energy
SWD (K)	0.003	0.044	0.002	0.002	0.037	0.001	0.011	0.083	0.010	0.016	0.099	0.014
SWD (Any)	0.045	0.146	0.044	0.003	0.044	0.002	0.003	0.038	0.002	0.001	0.031	0.000
Energy (K)	0.002	0.036	0.001	0.004	0.052	0.002	0.009	0.092	0.008	0.016	0.101	0.015
Energy (Any)	0.077	0.173	0.076	0.003	0.047	0.002	0.001	0.031	-0.000	0.001	0.033	-0.000
FM (K)	0.008	0.072	0.006	0.009	0.077	0.008	0.012	0.092	0.011	0.021	0.125	0.020
FM (Any)	0.034	0.120	0.032	0.014	0.098	0.012	0.012	0.094	0.011	0.013	0.094	0.012

OOD Targets

	$K = 10$			$K = 100$			$K = 1,000$			$K = 10,000$		
Model	MMD	SWD	Energy	MMD	SWD	Energy	MMD	SWD	Energy	MMD	SWD	Energy
SWD (K)	0.911	0.642	0.910	0.147	0.242	0.145	0.037	0.138	0.036	0.020	0.110	0.019
SWD (Any)	0.147	0.219	0.146	0.004	0.046	0.003	0.003	0.039	0.001	0.001	0.031	0.000
Energy (K)	0.942	0.651	0.941	0.127	0.227	0.126	0.036	0.143	0.035	0.020	0.111	0.019
Energy (Any)	0.171	0.237	0.169	0.004	0.048	0.002	0.001	0.035	0.000	0.001	0.033	-0.000
FM (K)	0.930	0.649	0.929	0.123	0.221	0.121	0.034	0.134	0.032	0.027	0.134	0.025
FM (Any)	0.071	0.168	0.070	0.015	0.102	0.014	0.013	0.096	0.012	0.012	0.090	0.011

F. MNIST-Colors image benchmark

This appendix describes the MNIST-Colors image experiment, which serves as a sanity check that the qualitative behavior observed in the MVN/GMM scaling experiments (Appendix E.7) persists on simple image data. Generator families and standard distributional metrics are defined in Appendices C and D.

F.1. Dataset

We construct a dataset of image distributions by taking grayscale MNIST digits (LeCun et al., 2010) and applying an RGB "style" via elementwise multiplication with a color vector $c \in [0, 1]^3$. Each distribution is represented by a set of $n = 64$ colored images of shape $3 \times 28 \times 28$. We study scaling with K by restricting training to $K \in \{10, 100, 1{,}000, 10{,}000\}$ unique colors (with repetition to keep the total number of training sets fixed). We evaluate both in-distribution (IID) targets that use colors from the training set and out-of-distribution (OOD) targets that use held-out colors.

F.2. Models

Encoders. For any-to-any (STC) models, we use a convolutional distribution encoder that processes each image with a shared CNN and aggregates across the set via mean-pooled message passing, outputting $z \in \mathbb{R}^{64}$. For the K-to-K baseline, we use a learned embedding lookup table with K embeddings.

Generators. All MNIST-Colors generators use a context-conditioned U-Net backbone. For the deterministic SWD and Energy regressors (Appendix C), the U-Net is conditioned on the concatenated embeddings $(z_{\text{src}}, z_{\text{tgt}})$ and predicts transported images $\hat{y} = \mathcal{T}_\theta(x \mid z_{\text{src}}, z_{\text{tgt}})$. For flow matching (FM), we use the same backbone as a time-conditional velocity field $v_\theta(x_t, t, z_{\text{src}}, z_{\text{tgt}})$ with $\sigma = 0.5$.

Training. Unless stated otherwise, we use latent dimension 64, set size 64, batch size 32, learning rate 2×10^{-4}, and train for 200 epochs.

F.3. Metrics and results

We report SWD and MMD-RBF (Appendix D) computed on flattened pixels, as well as a color error metric defined as the mean squared error between the per-image mean RGB vectors of generated and target samples (**Color MSE**). Full results are shown in Table 11.

Table 8. Distributional distances (↓) for GMM experiments across K unique distributions. We report MMD, SWD, and Energy distance.

IID Targets

	$K = 10$			$K = 100$			$K = 1,000$			$K = 10,000$		
Model	MMD	SWD	Energy	MMD	SWD	Energy	MMD	SWD	Energy	MMD	SWD	Energy
SWD (K)	0.004	0.115	0.001	0.006	0.127	0.003	0.011	0.148	0.008	0.020	0.186	0.017
SWD (Any)	0.948	0.866	0.945	0.056	0.271	0.053	0.019	0.163	0.016	0.009	0.127	0.006
Energy (K)	0.004	0.114	0.000	0.005	0.131	0.002	0.009	0.154	0.005	0.020	0.208	0.017
Energy (Any)	0.937	0.947	0.934	0.043	0.276	0.040	0.010	0.159	0.007	0.009	0.149	0.005
FM (K)	0.045	0.302	0.042	0.054	0.352	0.051	0.062	0.366	0.059	0.055	0.339	0.052
FM (Any)	0.284	0.564	0.280	0.060	0.308	0.057	0.044	0.280	0.041	0.037	0.269	0.034

OOD Targets

	$K = 10$			$K = 100$			$K = 1,000$			$K = 10,000$		
Model	MMD	SWD	Energy	MMD	SWD	Energy	MMD	SWD	Energy	MMD	SWD	Energy
SWD (K)	0.787	0.882	0.784	0.337	0.567	0.334	0.169	0.408	0.166	0.113	0.341	0.110
SWD (Any)	0.764	0.851	0.760	0.058	0.271	0.054	0.015	0.149	0.011	0.010	0.129	0.006
Energy (K)	0.786	0.884	0.783	0.335	0.564	0.332	0.171	0.409	0.168	0.109	0.345	0.106
Energy (Any)	1.395	1.109	1.392	0.041	0.269	0.038	0.010	0.154	0.007	0.008	0.142	0.005
FM (K)	0.782	0.858	0.779	0.350	0.592	0.348	0.187	0.465	0.184	0.124	0.401	0.121
FM (Any)	0.267	0.607	0.264	0.059	0.304	0.056	0.046	0.284	0.043	0.039	0.278	0.036

Table 9. Distributional distances (↓) for MVN semi-supervised experiments. IID: $\|\mu\|_\infty \leq 2.5$, OOD: $\|\mu\|_\infty > 2.5$.

		IID			**OOD**		
Generator	**Method**	MMD	SWD	Energy	MMD	SWD	Energy
Flow Matching	Supervised	0.000	0.003	-0.000	0.000	0.006	-0.000
	Semi-sup.	0.001	0.019	0.001	0.001	0.021	0.001
	Oracle	0.001	0.020	0.001	0.003	0.028	0.003
Energy	Supervised	0.000	0.003	-0.000	0.020	0.062	0.020
	Semi-sup.	0.001	0.021	0.000	0.003	0.033	0.002
	Oracle	0.001	0.021	0.000	0.002	0.030	0.002
SWD	Supervised	0.000	0.003	-0.000	0.017	0.052	0.017
	Semi-sup.	0.001	0.024	0.001	0.002	0.034	0.002
	Oracle	0.001	0.024	0.001	0.003	0.036	0.003

The MNIST-Colors results mirror the Gaussian experiments: at small K, K-to-K baselines can perform competitively on IID targets but degrade on OOD targets, while any-to-any models improve OOD performance consistently across generator families. As K increases, the gap between the two regimes narrows, reflecting the reduced need to interpolate beyond the training set.

G. Experimental details for scRNA-seq batch effect transfer

This appendix provides experimental details for the scRNA-seq batch effect transfer experiment in Section 5.2. We describe the preprocessing pipeline, model configurations, and evaluation protocol. Generator families and metrics are defined in Appendices C and D.

G.1. Data preprocessing

We use the murine pancreas scRNA-seq atlas data collected in Hrovatin et al. (2023). We use a preprocessed version of the dataset distributed by `cellxgene`[1]. We use the top $d = 10$ principal components for all downstream analysis. These 10 PCs are standardized to have unit variance and zero mean.

[1] https://cellxgene.cziscience.com/collections/296237e2-393d-4e31-b590-b03f74ac5070

Table 10. Distributional distances (↓) for GMM semi-supervised experiments. IID: $\|\mu\|_\infty \leq 2.5$, OOD: $\|\mu\|_\infty > 2.5$.

		IID			OOD		
Generator	**Method**	MMD	SWD	Energy	MMD	SWD	Energy
Flow Matching	Supervised	0.000	0.015	0.000	0.001	0.022	0.001
	Semi-sup.	0.003	0.032	0.002	0.011	0.052	0.011
	Oracle	0.003	0.032	0.002	0.005	0.041	0.005
Energy	Supervised	0.000	0.008	0.000	0.037	0.094	0.037
	Semi-sup.	0.001	0.028	0.001	0.009	0.061	0.008
	Oracle	0.002	0.029	0.001	0.005	0.044	0.005
SWD	Supervised	0.000	0.008	0.000	0.018	0.065	0.018
	Semi-sup.	0.003	0.036	0.003	0.013	0.070	0.013
	Oracle	0.003	0.036	0.003	0.009	0.059	0.009

Table 11. Distributional distances (↓) for MNIST-Colors across K unique training colors. We report SWD, MMD-RBF, and Color MSE.

IID Targets

	$K = 10$			$K = 100$			$K = 1,000$			$K =$	
Model	SWD	MMD	Color MSE	SWD	MMD	Color MSE	SWD	MMD	Color MSE	SWD	MM
SWD (K-to-K)	0.0472	0.3746	0.0011	0.0425	0.3395	0.0009	0.0444	0.3549	0.0010	0.0442	0.35
SWD (Any-to-Any)	0.0415	0.2987	0.0014	0.0400	0.3138	0.0011	0.0404	0.2956	0.0012	0.0414	0.31
Energy (K-to-K)	0.0426	0.3225	0.0013	0.0387	0.2905	0.0011	0.0408	0.3104	0.0012	0.0406	0.31
Energy (Any-to-Any)	0.0392	0.2549	0.0013	0.0378	0.2630	0.0011	0.0400	0.2833	0.0012	0.0404	0.29
Flow Matching (K-to-K)	0.0763	1.2057	0.0017	0.0634	0.8942	0.0012	0.0656	0.9301	0.0014	0.0657	0.93
Flow Matching (Any-to-Any)	0.0678	0.9243	0.0015	0.0575	0.7122	0.0012	0.0595	0.7879	0.0014	0.0563	0.65

OOD Targets

	$K = 10$			$K = 100$			$K = 1,000$			$K =$	
Model	SWD	MMD	Color MSE	SWD	MMD	Color MSE	SWD	MMD	Color MSE	SWD	MM
SWD (K-to-K)	0.0646	0.9201	0.0021	0.0493	0.4626	0.0012	0.0459	0.3825	0.0011	0.0459	0.37
SWD (Any-to-Any)	0.0612	0.8022	0.0016	0.0430	0.3481	0.0012	0.0409	0.3012	0.0012	0.0427	0.33
Energy (K-to-K)	0.0638	0.8685	0.0019	0.0463	0.4114	0.0013	0.0424	0.3352	0.0012	0.0422	0.32
Energy (Any-to-Any)	0.0601	0.7550	0.0018	0.0413	0.3037	0.0012	0.0408	0.2912	0.0012	0.0417	0.30
Flow Matching (K-to-K)	0.0793	1.3419	0.0018	0.0692	1.0138	0.0014	0.0677	0.9777	0.0014	0.0659	0.90
Flow Matching (Any-to-Any)	0.0748	1.2180	0.0018	0.0601	0.7541	0.0013	0.0610	0.8206	0.0014	0.0560	0.63

Train-test split. In order to induce a distribution shift at test-time, we hold out all mice of one condition. We hold out three donor mice at the 2 year age, leaving 53 remaining donor mice for the training dataset.

Dataset sampling. For each donor in the split, we sample sets of $n = 128$ cells uniformly at random. At training time, for each source donor, a random target donor is selected from the training set, and sets are sampled from both, yielding random source-target sets $(S_i^{\text{src}}, S_i^{\text{tgt}})$. This corresponds to STC (any-to-any) training in the terminology of Appendix C.

G.2. Model configurations

We evaluate the three standard continuous-data generator families described in Appendix C: flow matching (FM), energy/MMD regression (Energy), and sliced Wasserstein regression (SWD). For each generator type, we compare two encoder architectures: (i) a *distribution encoder* (as in DCT) that maps each sampled cell set to an embedding, and (ii) a *one-hot encoder* that learns a fixed embedding vector for each donor in the training set (a K-conditioned baseline). Unless stated otherwise, the distribution encoder uses the standard architecture described in Appendix C.2.

For the one-hot encoder at test time, we assign each test donor to its nearest training donor based on Euclidean distance between donor centroids (computed as the mean of all cells in PCA space for that donor). The model then uses the learned embedding of this nearest training donor.

Generator	Encoder	Energy ↓	SWD ↓	MMD-RBF ↓
SWD	Any-to-any	0.1167 ± 0.0119	0.3161 ± 0.0326	0.2334 ± 0.0239
	K-to-K	0.3411 ± 0.0351	0.4865 ± 0.0317	0.6822 ± 0.0702
Energy	Any-to-any	0.0729 ± 0.0090	0.3551 ± 0.0354	0.1459 ± 0.0181
	K-to-K	0.3623 ± 0.0327	0.5998 ± 0.0344	0.7245 ± 0.0655
FM	Any-to-any	0.1480 ± 0.0157	0.4532 ± 0.0361	0.2960 ± 0.0315
	K-to-K	0.4235 ± 0.0540	0.5448 ± 0.0425	0.8470 ± 0.1080
scVI		0.8056 ± 0.0643	0.6190 ± 0.0630	1.6112 ± 0.1286

Table 12. Distributional metrics for batch effect transfer between held-out donors (mean ± standard error).

Baselines. We compare against scVI (Lopez et al., 2018), a deep generative model for scRNA-seq that learns a latent representation while correcting for batch effects. We train scVI on the three held-out donors, using the donor identity as the batch condition. We use a model with 10 latent dimensions and 2 hidden layers, and 128 hidden units per layer. We train for 1000 epochs. We evaluate its ability to transform source distributions to match target distributions via the `transform_batch` functionality, which changes the batch condition passed to the decoder. The resulting normalized expression is then matched to its nearest neighbor in the target dataset.

Evaluation protocol. For each test donor pair (source, target), we sample $n = 10$ independent source-target set pairs. For our models, we generate predictions by sampling from the learned generator conditioned on the source distribution and target embedding. For scVI, we transform the source cells to the target batch and use the observed nearest neighbor for each prediction. We then compute metrics between the predicted and ground truth target distributions.

Metrics. We report the standard distributional metrics defined in Appendix D: energy distance, sliced Wasserstein distance, and MMD with an RBF kernel. Metrics are computed between generated and ground-truth target distributions for each sampled pair, aggregated across all test donor combinations and reported as mean ± standard error.

G.3. Full results across metrics

In Table 1, we show MMD-RBF for all evaluated models. In Table 12, we show all three evaluated metrics.

H. Drug perturbation prediction on organoids (Trellis)

This appendix provides additional details for the Trellis organoid drug-perturbation benchmarks in Section 6.2. We use the Trellis single-cell mass cytometry dataset (Zapatero et al., 2023), representing each cell by 43 protein abundance features and forming 927 matched control–treatment population pairs across 10 patients and 11 treatments. Each population is represented at training time by a set of $n = 256$ cells sampled uniformly at random.

We evaluate on two replicate-holdout splits (`replicas-1`, `replicas-2`) and three patient-holdout splits (`pdo21`, `pdo27`, `pdo75`). Models use the standard continuous-data generators (SWD/Energy/FM; Appendix C) and are evaluated with the standard metrics (Appendix D). We report three conditioning regimes: a supervised source-conditioned (SC) baseline (MFM) that conditions on treatment identity, a semi-supervised source–target-conditioned (STC) model with a post-hoc ridge predictor for the target embedding, and an oracle STC evaluation that conditions on the true target embedding. Unless stated otherwise, the distribution encoder uses the standard architecture described in Appendix C.2.

Key hyperparameters. Unless otherwise stated, we use embedding dimension 256, hidden dimension 256, batch size 4, learning rate 2×10^{-4}, and train for 20,000 epochs.

Table 13. Distributional metrics (↓). Mean ± standard deviation reported for IID (replicate holdout) and OOD (patient holdout) settings.

Generator	Regime	Energy	SWD	MMD-RBF
		IID (Replicate Holdout)		
SWD	Supervised (SC)	0.0897 ± 0.0332	0.1266 ± 0.0256	0.0064 ± 0.0022
	Semi-supervised (STC)	0.1305 ± 0.0305	0.1524 ± 0.0185	0.0094 ± 0.0018
	Oracle (STC)	0.0154 ± 0.0005	0.0633 ± 0.0009	0.0010 ± 0.0000
Energy	Supervised (SC)	0.0779 ± 0.0290	0.1199 ± 0.0209	0.0054 ± 0.0019
	Semi-supervised (STC)	0.1442 ± 0.0520	0.1588 ± 0.0297	0.0103 ± 0.0032
	Oracle (STC)	0.0169 ± 0.0026	0.0659 ± 0.0037	0.0011 ± 0.0002
FM	Supervised (SC)	0.1431 ± 0.0377	0.1621 ± 0.0187	0.0101 ± 0.0023
	Semi-supervised (STC)	0.1450 ± 0.0380	0.1602 ± 0.0168	0.0106 ± 0.0025
	Oracle (STC)	0.0415 ± 0.0034	0.0929 ± 0.0054	0.0030 ± 0.0003
		OOD (Patient Holdout)		
SWD	Supervised (SC)	0.3391 ± 0.0093	0.2452 ± 0.0078	0.0259 ± 0.0008
	Semi-supervised (STC)	0.2580 ± 0.0403	0.2103 ± 0.0196	0.0197 ± 0.0022
	Oracle (STC)	0.0346 ± 0.0098	0.0819 ± 0.0066	0.0026 ± 0.0009
Energy	Supervised (SC)	0.3932 ± 0.0258	0.2622 ± 0.0074	0.0298 ± 0.0019
	Semi-supervised (STC)	0.2503 ± 0.0556	0.2091 ± 0.0246	0.0191 ± 0.0049
	Oracle (STC)	0.0315 ± 0.0016	0.0808 ± 0.0028	0.0022 ± 0.0002
FM	Supervised (SC)	0.2969 ± 0.0209	0.2247 ± 0.0089	0.0227 ± 0.0006
	Semi-supervised (STC)	0.2668 ± 0.0347	0.2138 ± 0.0167	0.0208 ± 0.0018
	Oracle (STC)	0.0705 ± 0.0214	0.1185 ± 0.0200	0.0053 ± 0.0013

H.1. Full results across metrics

H.2. scGen baseline

	Energy	SWD	RBF
IID	$0.2924_{\pm 0.0881}$	$0.2189_{\pm 0.0383}$	NA
OOD	$0.3278_{\pm 0.0345}$	$0.2272_{\pm 0.0162}$	$0.0265_{\pm 0.0022}$

I. Experimental details for lineage-traced scRNA-seq

This appendix provides experimental details for the lineage-traced scRNA-seq forecasting experiment in Section 6.3. We describe dataset construction, model configurations, and evaluation. Generator families and metrics are defined in Appendices C and D.

I.1. Data preprocessing

We use the *in vitro* lineage-traced scRNA-seq dataset from Weinreb et al. (2020), which tracks hematopoietic stem cell differentiation over time points $t \in \{2, 4, 6\}$ days.

We download normalized count matrices, metadata, gene names, and clone assignment matrices from the paper data repository[2]. We discard all cells which have not been assigned to a clone.

Gene expression preprocessing. Following standard scRNA-seq preprocessing pipelines (Heumos et al., 2023), we apply:

1. Normalization to 10^4 counts per cell

[2] https://kleintools.hms.harvard.edu/paper_websites/state_fate2020/

2. $\log(x+1)$ transform
3. Feature selection of top 10,000 highly variable genes
4. Unit variance, zero mean rescaling per gene with maximum value clipped at 10
5. PCA to $d = 50$ principal components

Downstream analysis is done on these 50 principal components.

Clone set construction. For each (clone, timepoint) pair, we partition cells into sets of size $n = 100$ with minimum $n_{\min} = 3$ cells per set. Sets smaller than n are padded by sampling with replacement from the same (clone, timepoint) group. This yields N sets $\{S_i\}_{i=1}^{N}$.

Dataset sampling (supervised). We construct source-target pairs $(S_i^{\text{src}}, S_i^{\text{tgt}})$ where both sets share the same clone identity but differ by $\Delta t = 2$ days (i.e., $t \in \{2, 4\}$ paired with $t + 2 \in \{4, 6\}$). This results in 1256 clones observed at consecutive timepoints. These clones are split into train/test with ratio 1:1. This split is at the level of clones, not cells – each clone appears in only part of the split.

Dataset sampling (semi-supervised). For the semi-supervised setting, we additionally construct unstructured pairs by matching source sets from clones without complete targets to random target sets at the correct timepoint. During training, each batch samples from the true paired distribution with probability $p = 0.25$ and from the random pairing with probability $1 - p = 0.75$. Clone splits are identical.

I.2. Model configurations

We evaluate the three standard continuous-data generator families described in Appendix C: flow matching (FM), energy/MMD regression (Energy), and sliced Wasserstein regression (SWD). For each generator type, we compare three regimes (as in the main text): a supervised source-conditioned (SC) baseline trained only on paired clones; a semi-supervised source–target-conditioned (STC) model trained on a mixture of true pairs and random early→late pairings and evaluated using a post-hoc ridge predictor for the target embedding; and an oracle STC evaluation that conditions on the true target embedding. Unless stated otherwise, the distribution encoder uses the standard architecture described in Appendix C.2.

For the semi-supervised setting, we fit a linear predictor post-hoc to map source embeddings to target embeddings. Specifically, we encode all training source and target distributions using the semi-supervised encoder, then fit a ridge regression model with 5-fold cross-validation over regularization values $\alpha \in [10^{-6}, 10^{6}]$ to predict target embeddings from source embeddings.

Metrics. We report the standard distributional metrics defined in Appendix D: energy distance (Energy), sliced Wasserstein distance (SWD), and MMD with an RBF kernel (MMD-RBF). Metrics are computed between generated and ground-truth target distributions for each test clone, with results reported as mean $\pm$ standard error across test clones.

I.3. Full results across metrics

In Table 6.3 we show MMD-RBF for transport across all evaluated models. In Table 14, we show results across all three evaluated metrics. Regardless of the distributional metric, results remain qualitatively similar.

J. TCR Repertoire Forecasting Experiments

This appendix provides detailed descriptions of the dataset, model architectures, and evaluation protocols for the T-cell receptor repertoire forecasting experiments presented in Section 6.4. We describe the data processing pipeline, the two discrete sequence generators we employ, and explain architectural differences that may account for their divergent performance. Distributional evaluation metrics are defined in Appendix D and are computed here in ESM2 embedding space.

Generator	Regime	Energy ↓	SWD ↓	MMD-RBF ↓
SWD	Supervised (SC)	4.936 ± 0.133	1.679 ± 0.023	9.872 ± 0.267
	Semi-supervised (STC)	4.333 ± 0.108	1.575 ± 0.020	8.666 ± 0.215
	Oracle (STC)	1.494 ± 0.025	0.953 ± 0.011	2.988 ± 0.049
Energy	Supervised (SC)	4.709 ± 0.127	1.650 ± 0.023	9.419 ± 0.255
	Semi-supervised (STC)	4.269 ± 0.103	1.575 ± 0.020	8.538 ± 0.205
	Oracle (STC)	1.440 ± 0.022	0.951 ± 0.011	2.881 ± 0.045
FM	Supervised (SC)	7.049 ± 0.139	2.239 ± 0.025	14.099 ± 0.278
	Semi-supervised (STC)	4.834 ± 0.123	1.747 ± 0.023	9.668 ± 0.247
	Oracle (STC)	2.539 ± 0.052	1.309 ± 0.017	5.077 ± 0.104

Table 14. Distributional metrics for clonal distribution forecasting in held-out clones (mean ± standard error).

J.1. Data preprocessing

J.1.1. Source Data

We use longitudinal TCR repertoire sequencing data from COVID-19 patients collected by Schultheiß et al. (2020). The dataset contains TRB (T-cell receptor beta chain) CDR3 amino acid sequences from 37 patients, of whom 10 were profiled at multiple timepoints during disease progression and recovery. Each repertoire comprises thousands of unique CDR3 junction sequences, typically 10–20 amino acids in length.

J.1.2. Sequence Filtering

Raw CDR3 junction sequences are filtered to retain only those composed of the 20 standard amino acids (ACDEFGHIKLMNPQRSTVWY). Sequences containing ambiguous or non-standard residues are discarded. We truncate or pad sequences to a maximum length of 30 tokens (including special tokens), which accommodates the vast majority of TCR CDR3 sequences.

J.1.3. Train-Test Split

The train-test split is performed at the patient level to ensure that test evaluation reflects true generalization to unseen individuals. We partition the 10 multi-timepoint patients, holding out 3 patients (30%) for evaluation and training on the remaining 7. Single-timepoint patients are included only in training for the semi-supervised (any-to-any) model, where they serve as additional unpaired marginals that enrich the learned embedding space.

For the supervised (within-patient) model, only consecutive timepoint pairs $(t, t{+}1)$ from multi-timepoint patients are used for training. The semi-supervised model additionally forms cross-patient pairs, training on transitions from any earlier timepoint to any later timepoint across different patients. This pairing strategy encodes the inductive bias that immune repertoires evolve forward in time, while allowing the model to leverage the larger pool of unpaired data.

J.1.4. Tokenization

Each sequence is tokenized for both ESM2 and ProGen2 vocabularies and cached for efficient training. ESM2 tokenization follows the standard protocol with <cls> and <eos> special tokens, producing input sequences of the form ⟨cls⟩SEQUENCE⟨eos⟩⟨pad⟩$\cdots$. ProGen2 tokenization uses <|bos|> and <|eos|> delimiters with <|pad|> for padding.

J.2. Embedding Architecture

All models share a common distribution encoder that maps repertoires to fixed-dimensional embeddings. The encoder operates in two stages: a pretrained protein language model extracts per-sequence representations, which are then aggregated into a distribution embedding.

J.2.1. SEQUENCE ENCODER

We use ESM2 (Lin et al., 2023) with 8 million parameters as the backbone sequence encoder. For each sequence, we extract the final hidden states and apply mean pooling over non-padding positions to obtain a 320-dimensional sequence embedding. The ESM2 backbone is frozen during training to preserve its pretrained representations.

J.2.2. DISTRIBUTION ENCODER

Given a set of n sequence embeddings $\{h_1, \ldots, h_n\} \subset \mathbb{R}^{320}$, we apply a transformer-based aggregator following the architecture of Zaheer et al. (2017). The aggregator consists of:

1. An input projection $\texttt{Linear}(320, 256) + \texttt{SELU}$
2. Two self-attention layers with 4 heads each
3. Mean pooling across sequences
4. A latent projection $\texttt{Linear}(256, 128) + \texttt{SELU}$
5. ℓ_2 normalization of the final embedding

The output is a 128-dimensional distribution embedding $z \in \mathbb{R}^{128}$ that summarizes the entire repertoire.

J.3. Generator Architectures

We compare two discrete sequence generators that share the same conditioning interface but differ fundamentally in their generative mechanisms.

J.3.1. ESM2 DISCRETE FLOW MATCHING (DFM)

The DFM generator adapts the discrete flow matching framework of Gat & Lipman (2024) to protein sequence generation by repurposing ESM2 as a denoising backbone. Rather than training an autoregressive model, DFM learns to iteratively refine sequences from source to target through a learned flow over discrete tokens.

Architecture. We extend `EsmForMaskedLM` with conditioning modules that inject source and target distribution embeddings at two points:

1. **Input-level conditioning:** The concatenated latent $[z_{\text{src}}; z_{\text{tgt}}] \in \mathbb{R}^{64}$ is projected through a two-layer MLP ($\texttt{Linear}(64, 256) \to \texttt{GELU} \to \texttt{Linear}(256, 320)$) and added to the token embeddings before the transformer encoder. This ensures all 6 transformer layers process conditioned representations.
2. **Output-level conditioning:** A parallel projection is added to the hidden states after the transformer, before the language modeling head.

Time conditioning uses sinusoidal embeddings following the original transformer formulation, with the continuous time $t \in [0, 1]$ embedded into the model's hidden dimension and added to token embeddings.

Training. Given source sequence x^{src} and target sequence x^{tgt}, we sample $t \sim \text{Uniform}(0, 1)$ and construct an interpolated sequence x_t via Bernoulli masking: at each position, x_t equals x^{tgt} with probability t and x^{src} otherwise. The model predicts x^{tgt} from x_t using cross-entropy loss over all positions, including padding tokens. This allows the model to learn length changes (insertions and deletions) between source and target.

Sampling. Generation proceeds by discrete flow integration from $t = 0$ to $t = 1$ with step size $\Delta t = 0.05$. At each step, the model predicts a distribution over tokens at each position. The update uses a mixture formula:

$$p(x_{t+\Delta t}) = \mathbf{1}_{x_t} + \frac{\Delta t}{1 - t} \left(p_\theta(x^{\text{tgt}} \mid x_t, t) - \mathbf{1}_{x_t} \right) \tag{34}$$

where $\mathbf{1}_{x_t}$ is the one-hot encoding of the current sequence. Vocabulary constraints enforce that position 0 contains only `<cls>` and subsequent positions contain only amino acids, `<eos>`, or `<pad>`.

J.3.2. ProGen2 with Prefix Conditioning

The ProGen2 generator uses the pretrained ProGen2 language model (Nijkamp et al., 2023) with prefix-based conditioning for source-to-target generation.

Architecture. We use `progen2-medium` as the backbone. Conditioning is implemented by inserting a learned prefix embedding between source and target sequences:

1. The concatenated latent $[z_{\text{src}}; z_{\text{tgt}}]$ is projected through $\texttt{Linear}(64, 256) \to \texttt{GELU} \to \texttt{Linear}(256, d_{\text{model}})$
2. The resulting vector is inserted as a single prefix token between the source and target portions of the input
3. The attention mask is adjusted to allow the prefix to attend to source tokens and target tokens to attend to the prefix

Training. The input sequence is formed by concatenating source and target: ⟨bos⟩SOURCE⟨eos⟩⟨bos⟩TARGET⟨eos⟩. The prefix is inserted at the boundary. The model is trained with causal language modeling loss computed only on target tokens, ignoring source and prefix positions.

Sampling. Generation proceeds autoregressively: given the source sequence and predicted target embedding, the model generates target tokens one at a time until <eos> or maximum length. To encourage diversity when generating multiple samples, we add Gaussian noise with scale 0.1 to the latent embeddings.

J.4. Evaluation Protocol

All models are evaluated on the same task: given a patient's repertoire at time t, forecast the repertoire at time $t{+}1$. We evaluate only on consecutive timepoint pairs from held-out patients.

J.4.1. Embedding Computation

For each repertoire, we sample sequences uniformly at random and compute the distribution embedding using the trained encoder. At training time, we use 2048 sequences per repertoire to compute embeddings for the predictor.

J.4.2. Target Embedding Prediction

For the semi-supervised model, we train a ridge regression predictor on the delta $z_{t+1} = z_t + \Delta z, \Delta z = W z_t + b$ on consecutive pairs from training patients, using 2048 sequences per repertoire for embedding computation. Regularization strength is selected via leave-one-out cross-validation over a grid of 20 log-spaced values from 10^{-4} to 10^{4}.

J.4.3. Sequence Generation and Comparison

For each test pair:

1. Encode the current timepoint repertoire to obtain z_t
2. Form the target embedding $\hat{z}_{t+1}$ using either (i) a supervised within-patient baseline that sets $\hat{z}_{t+1} = 0$ (source-conditioned), (ii) a semi-supervised ridge predictor $\hat{z}_{t+1}$ (source–target-conditioned), or (iii) an oracle that uses the true target embedding z_{t+1}
3. Generate 2048 sequences conditioned on z_t and $\hat{z}_{t+1}$
4. Re-tokenize generated sequences for ESM2 and compute mean-pooled embeddings
5. Sample 2048 sequences from the actual next timepoint and compute their ESM2 embeddings
6. Compute distributional distances between generated and actual sequence embeddings

J.4.4. METRICS

We report the standard distributional metrics defined in Appendix D (energy distance, sliced Wasserstein distance, and MMD-RBF), computed in the ESM2 embedding space.

Results are averaged over all test pairs, with standard errors computed across pairs.

J.5. Discussion: Why ProGen2 Underperforms

The results in Table 4 show that the DFM generator substantially outperforms ProGen2, particularly in the semi-supervised setting. We hypothesize several factors contribute to this gap.

Autoregressive vs. iterative refinement. ProGen2 generates sequences left-to-right, committing to each token before seeing subsequent context. The DFM generator, by contrast, refines all positions simultaneously through iterative denoising. For the task of transforming one sequence into another (rather than generating de novo), iterative refinement may better preserve structural features of the source while adapting to the target distribution.

Conditioning mechanism. The DFM generator injects conditioning at both input and output levels of every transformer layer, ensuring the distributional signal permeates the entire network. ProGen2's prefix conditioning provides a single conditioning vector that the model must propagate through causal attention. This architectural difference may limit ProGen2's ability to leverage the target embedding for precise distributional control.

J.6. Hyperparameter Summary

Table 15 summarizes the key hyperparameters for reproducibility.

Table 15. Hyperparameters for TCR repertoire forecasting experiments.

Hyperparameter	**DFM**	**ProGen2**
Dataset		
Sequences per repertoire (training)	2048	2048
Maximum sequence length	30	30
Test fraction (multi-TP patients)	0.3	0.3
Encoder		
ESM2 backbone	`esm2_t6_8M`	`esm2_t6_8M`
Distribution embedding dim	128	128
Self-attention layers	2	2
Attention heads	4	4
Generator		
Backbone	`esm2_t6_8M`	`progen2-medium`
Conditioning dimension	256	256
Latent input dimension	32	32
Temperature	1.0	1.0
Sample step size (Δt)	0.05	—
Training		
Batch size	1	1
Learning rate	2×10^{-5}	2×10^{-5}
Optimizer	Adam	Adam
Evaluation		
Sequences per repertoire	2048	2048
Predictor training samples	2048	2048
Predictor regularization	RidgeCV	RidgeCV

J.6.1. RESULTS

Table 16 records results across all metrics.

Table 16. TCR repertoire prediction performance across conditioning strategies (mean ± standard error). DFM shows meaningful conditioning (Oracle ≫ Predictor ≫ Within-patient), while ProGen2 shows little conditioning effect due to encoder collapse (all embeddings have cosine similarity > 0.998).

Generator	Regime	Energy ↓	SWD ↓	MMD-RBF ↓
ProGen2	Within-patient (SC)	0.0682 ± 0.0103	0.4015 ± 0.0118	0.0588 ± 0.0085
	Predictor (STC)	0.0681 ± 0.0104	0.4012 ± 0.0118	0.0587 ± 0.0085
	Oracle (STC)	0.0676 ± 0.0114	0.3856 ± 0.0148	0.0567 ± 0.0093
DFM	Within-patient (SC)	0.0684 ± 0.0120	0.3895 ± 0.0147	0.0579 ± 0.0100
	Predictor (STC)	0.0262 ± 0.0050	0.2778 ± 0.0152	0.0215 ± 0.0043
	Oracle (STC)	0.0090 ± 0.0015	0.2199 ± 0.0217	0.0078 ± 0.0013

J.7. Can we close the semi-supervised–oracle gap?

In the TCR setting, the gap between the *predictor* and *oracle* regimes in Table 16 reflects how well we can predict the next-timepoint embedding z_{t+1} from the current embedding z_t using only the limited paired longitudinal data. The main results use a ridge predictor (chosen for stability in low-data regimes). We also evaluated more expressive nonlinear predictors, but on the patient-holdout split these did not improve performance and often degraded it, consistent with overfitting and distribution shift across individuals.

Latent predictor	Energy ↓	SWD ↓	MMD-RBF ↓
Ridge (linear)	$0.2902_{\pm.0090}$	$1.6447_{\pm.0184}$	$0.2619_{\pm.0104}$
MLP	$0.2902_{\pm.0091}$	$1.6455_{\pm.0205}$	$0.2652_{\pm.0103}$
Random Forest	$0.2901_{\pm.0090}$	$1.6461_{\pm.0182}$	$0.2623_{\pm.0104}$

Table 17. Closing the semi-supervised–oracle gap on TCR forecasting. Each row uses the same trained STC generator but varies how the target embedding is predicted.