scNODE : Generative Model for Temporal Single Cell Transcriptomic Data Prediction

Jiaqi Zhang[1], Erica Larschan,[2,3] Jeremy Bigness[2,4], and Ritambhara Singh[1,2,4]

[1] Department of Computer Science, Brown University
[2] Center for Computational Molecular Biology, Brown University
[3] Department of Molecular Biology, Cell Biology and Biochemistry, Brown University
[4] Co-corresponding authors
{jeremy_bigness, ritambhara}@brown.edu

Abstract. Temporal measurements using single-cell RNA sequencing (scRNA-seq) technology enable the study of gene expression programs of individual cells for normal developmental processes or diseased physiological states. However, due to the labor, expense, and technical challenges associated with these experiments, researchers are only able to profile gene expression at discrete and sparsely spaced time points. This results in information loss between consecutive discrete time points and impedes the downstream developmental analysis. To address this problem, we propose scNODE , an end-to-end model that can simulate and predict realistic *in silico* single-cell samples at any time point to enable temporal downstream analyses. Our method integrates a variational autoencoder (VAE) with neural ordinary differential equations (ODEs) to predict gene expression in a continuous and non-linear latent space. Importantly, scNODE adds a regularization term to integrate overall dynamics of cell developments to the latent space, such that the learned latent representation is informative and interpretable. Our evaluations on six real-world scRNA-seq datasets show that scNODE achieves higher predictive performance than state-of-the-art methods. We further demonstrate that scNODE learns an interpretable latent space, and helps recapitulate cell-type developmental trajectories.
Availability: The data and code are publicly available at https://github.com/rsinghlab/scNODE.

Keywords: scRNA-seq · generative model · data prediction

1 Introduction

A fundamental challenge in biology is understanding how the gene expression programs of cellular populations change over time due to healthy developmental processes or pathological states [46,23]. With the advent of high-throughput single-cell RNA sequencing (scRNA-seq), researchers now routinely profile gene expression at single-cell resolution [29,10]. Temporal recording of scRNA-seq data at multiple timepoints allows for studying developmental mechanisms at the single-cell level, revealing substantial heterogeneity in gene expression even among cells within the same tissue type and at the same developmental stage [12,20]. Furthermore, downstream analyses of these experiments enable researchers to understand how biological processes unfold within cellular populations, such as by inferring cellular trajectories, identifying cell fate transitions, and characterizing differentially expressed genes [49,39].

However, time-series scRNA-seq experiments have significant limitations that impede inference and downstream analysis. Due to the substantial time and resources involved in carrying out such experiments, researchers generally only profile gene expression at discrete and sparsely-spaced timepoints [20]. Since it is not feasible to make continuous-time observations, this results in information loss between consecutive discrete timepoints. Moreover, gene expression for certain cell types is challenging to profile at late stages of biological development. For example, cellular senescence is a complex process that limits cell division and growth as a response to environmental factors or intrinsic cues. It is difficult to detect senescent cells after the senescence program has been triggered [18]. Therefore, we have limited profiles of senescent cells, which are vital for aging and therapy, outside the time range of experimental measurements. These missing data pose challenges for the application of data-driven models, especially if the end goal of a study is to use time series data to make inferences about developmental biological processes. Therefore, it is crucial to develop a model that can simulate and predict realistic *in silico* samples at any timepoint, whether within or beyond the measured time intervals, to enable temporal downstream analyses (Fig. 1**A-B**).

Single-cell data prediction has been investigated in previous studies [9] through various generative models. Marouf et al. [41] use generative adversarial networks (GAN) to predict gene expression data for specific cell types. Other recent works [69,50,37,53] use statistical models to fit characteristics of observed single-cell data and use the learned information to simulate real-world data. However, these methods are designed for single-cell data at a steady state and do not consider developmental dynamics. In addition, RNA velocity [35,5,4] is a category of approaches to uncover developmental trends and underlying kinetics of gene expression. Despite their success, they have inevitable limitations on gene expression prediction because they utilize pseudotimes [58] which do not model how cells differentiate with respect to physical time, use linear ODEs [42] that fail to capture complex gene expression dynamics, or have to use RNA velocity pre-computed from gene expression [13] that causes circular analysis [34] and double dipping [2] problems. Hence, all these methods are not feasible for predicting temporal single-cell data.

Recently, a few generative methods have introduced procedures to model cell differentiation trajectory and predict unmeasured timepoints. For example, Waddington-OT [52] interpolates between two timepoints by predicting cell-cell coupling by solving the optimal transport (OT) problem. TrajectoryNet [55] and MIOFlow [28] utilize continuous normalizing flows or neural ordinary differential equations (ODEs) [11] to model cellular trajectories and interpolate between static gene expression snapshots of two timepoints. However, these methods cannot predict future timepoints because they require gene expression measurements at two endpoints to learn cellular developments at intermediate times. To overcome this issue, PRESCIENT [68] learns the cellular differentiation landscape using neural ODEs, and cannot only interpolate between measured timepoints but extrapolate beyond them. Due to the high sparsity of scRNA-seq data in the high-dimensional gene space [36], this work applies principal component analysis (PCA) to reduce dimensionality to a low-dimensional space. Most interpolating models, including TrajectoryNet and MIOFlow, also adopt PCA for dimensionality reduction. But multiple studies [19,66,57] have found that the low-dimensional representations of scRNA-seq expression obtained from PCA may hinder subsequent analyses because PCA is a linear dimensionality reduction approach that cannot capture complex relationships embedded in highly heterogeneous single-cell data. For example, PCA has the issue of overcrowding representation [57,33] where cells of different types are poorly separated in PCA space, and distinctions between different cell types are lost. Moreover, the PCA latent space is computed from measured timepoints and fixed during modeling cellular dynamics, such that it cannot account for cell variabilities at unmeasured timepoints as they are unseen to

PCA computation, which limits the accuracy of model predictions. Overall, existing generative models either have limited capabilities for making extrapolations or use a less effective dimensionality reduction approach.

To address these limitations, we propose single-cell neural ODE (`scNODE`), which integrates a variational autoencoder (VAE) [32] with neural ODE [11] to predict gene expression between timepoints (interpolation), including multiple consecutive steps, and at future timepoints (extrapolation) based on non-linear and continuous latent space (Fig. 1**C**). First, `scNODE` uses VAE to obtain a latent space where global cellular structures are retained. The VAE model has been adopted in several novel studies [38,24] and has shown superior performance relative to PCA in accurately and efficiently computing low-dimensional representations of heterogeneous scRNA-seq data. `scNODE` models differential dynamics in the latent space and predicts single-cell gene expression for unmeasured timepoints with neural ODE. `scNODE` adopts a additional regularization term to integrate cellular development in the latent space, such that the final latent space is fine-tuned by the overall dynamics, improving its generalizability in predicting accurate gene expression of cells for unobserved timepoints. Importantly, this regularizer results in an informative and interpretable latent space that can assist temporal downstream analysis. We evaluate `scNODE` on six real-world scRNA-seq datasets and show that: (1) `scNODE` achieves higher predictive performance than state-of-the-art methods, indicating its superiority in data prediction, (2) `scNODE` learns an interpretable latent space, which enables identifying key driver genes as well as conducting *in silico* perturbation analyses, and (3) `scNODE` accurately recapitulates cell type developmental trajectories.

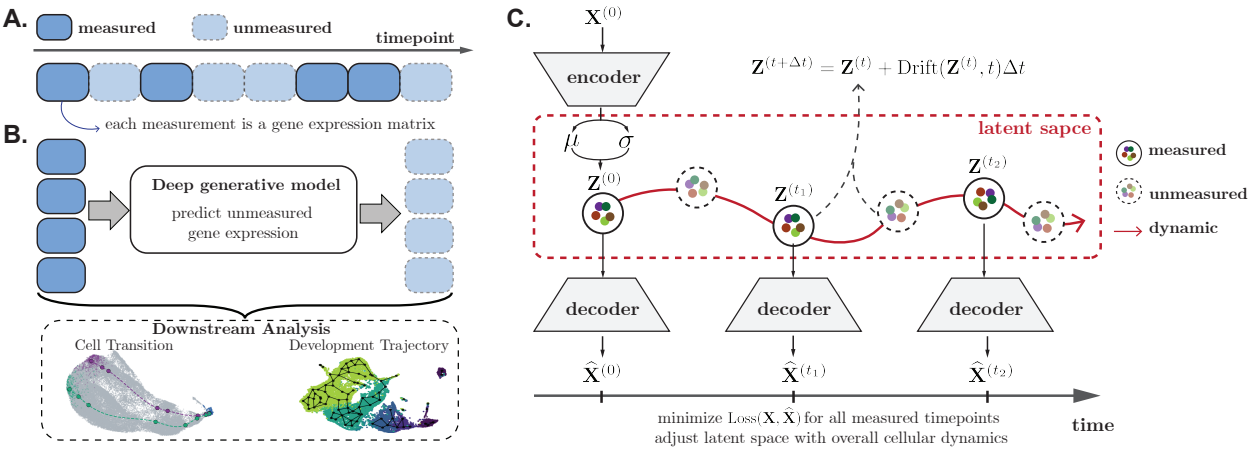

Fig. 1. Model overview. (**A**) The measurements in temporally recorded single-cell data are sparse and unaligned, such that some timepoints are not measured and measurements at different timepoints contain different sets of cells. The data sets in this study sample cellular populations at uniformly spaced time points, although uniform spacing is not a requirement (as depicted in this graphic). (**B**) `scNODE` is used to predict expression at unmeasured timepoints. Predictions can be used to analyze cell state transitions and developmental trajectories. (**C**) `scNODE` uses VAEs to find an optimal latent space representation, and then uses this representation to solve neural ODEs to predict gene expression at any timepoint $t = t_1, t_2, \cdots$ starting from the first timepoint $t = 0$. Optimal models are fitted by minimizing the Wasserstein distance between observations and predictions at measured timepoints. `scNODE` also adjusts latent space with the overall cellular dynamic.

2 Method

Notation : Let $\mathbf{X}^{(t)} \in \mathbb{R}^{n_t \times p}$ denote the gene expression matrix at the t-th time point of n_t cells by p genes. Given gene expression measurements at observed timepoints $\{\mathbf{X}^{(t)}\}_{t \in \mathcal{T}}$, our goal is to predict gene expression at any timepoint. Here, $\mathcal{T} \subseteq \{0, 1, 2, \cdots\}$ denotes the observed timepoint indices. Note that we assume the same set of highly variable genes (HGV) is measured at each time point, but due to the destruction of cells during scRNA experiments, a different set of cells is measured at each time point. Furthermore, let $\mathbf{Z}^{(t)} \in \mathbb{R}^{n_t \times d}$ denote the d-dimensional ($d < p$) latent variable at the t-th time point. Given a vector $\mathbf{x} \in \mathbb{R}^p$, let $\|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^{p} \mathbf{x}_i^2}$ denote the ℓ_2 norm.

2.1 `scNODE` uses VAE to learn complex latent space for cells at observed timepoints

`scNODE` uses a variational autoencoder (VAE) to learn a low-dimensional representation of the single-cell gene expression measurements at the observed timepoints used for training. A VAE is a neural network-based

generative model that maps high-dimensional data to a low-dimensional representation and is widely used in many single-cell studies [24,54,60]. VAE is composed of two neural networks: (1) the encoder network Enc_ϕ maps expression from gene space $\mathbb{R}^p$ to a low-dimensional latent space $\mathbb{R}^d$, parameterized by a Gaussian distribution ($\mathcal{N}(\mu, \sigma)$), and (2) the decoder network Dec_θ maps samples in the latent space to gene space in the opposite direction to reconstruct the input. Specifically, given the expression matrix $\mathbf{X}$, VAE predicts reconstructions of inputs through

$$\begin{gathered}\mu, \sigma = \text{Enc}_\phi(\mathbf{X}) \qquad \mathbf{Z} \sim \mathcal{N}(\mu, \sigma) \qquad \widehat{\mathbf{X}} = \text{Dec}_\theta(\mathbf{Z}) \\ \text{with} \quad \mathbf{X}, \widehat{\mathbf{X}} \in \mathbb{R}^{n \times p} \text{ and } \mathbf{Z} \in \mathbb{R}^{n \times d}.\end{gathered} \tag{1}$$

The encoder and decoder networks are parameterized by ϕ and θ, correspondingly. The benefit of using VAEs for single-cell data over PCA is that the non-linearity of neural networks can more effectively capture complex global cell relationships. Therefore, `scNODE` first pre-trains a VAE to obtain a latent space that captures the gene expression of single cells at all available training time points. Specifically, `scNODE` trains the VAE component to perform multi-timepoint modeling by inputting the gene expression of all concatenated cells $\mathbf{X}_{\text{ALL}} = \text{CONCAT}\left(\mathbf{X}^{(t)} \mid t \in \mathcal{T}\right)$. Here, we concatenate all the cells of only the training timepoints to avoid data leakage (see more details in Section 3). Next, it minimizes a combination of the (1) mean squared error (MSE) between input gene expression and the reconstructed gene expression from the decoder and (2) the Kulback-Leibler (KL) divergence [14] between the latent distribution and a standard Gaussian prior:

$$\begin{gathered}\mathcal{L}_{\text{pre-train}} = \text{MSE}\left(\mathbf{X}_{\text{ALL}},\ \widehat{\mathbf{X}}_{\text{ALL}}\right) + \lambda \text{KL}\left(\mathcal{N}(\mu_{\text{ALL}}, \sigma_{\text{ALL}}),\ \mathcal{N}(0,1)\right) \quad \text{where} \\ \mu_{\text{ALL}}, \sigma_{\text{ALL}} = \text{Enc}_\phi(\mathbf{X}_{\text{ALL}}), \quad \mathbf{Z}_{\text{ALL}} \sim \mathcal{N}(\mu_{\text{ALL}}, \sigma_{\text{ALL}}), \quad \widehat{\mathbf{X}}_{\text{ALL}} = \text{Dec}_\theta(\mathbf{Z}_{\text{ALL}}).\end{gathered} \tag{2}$$

KL divergence is an information-theoretic quantity that quantifies the distance between two probability distributions. By using a Gaussian prior, the KL divergence forces the encoder to find latent representations of the cells that are spread isotropically about the center of the space, ensuring that these latent representations are well-separated. `scNODE` pre-trains the VAE component with Eq. 2, such that the encoder latent space preserves the structural relationships of all observed cells. Next, it models the cell differential dynamics in this latent space.

2.2 `scNODE` uses Neural ODE to model cell development dynamics

`scNODE` uses ordinary differential equations (ODEs) to model cell developmental dynamics in the latent space learned by the VAE component. An ODE is an equation describing how a quantity x changes with respect to an independent variable y, such that $\mathrm{d}x = f(x; y) \cdot \mathrm{d}y$ where function f represents the derivative. Therefore, we can use differential equations to model how gene expression changes with respect to time. However, for the high-dimensional data, finding the solution of the derivative function f through numerical methods is intractable and computationally expensive [31]. Therefore, recent studies adopt neural networks to approximate the derivative function and have proposed neural ODE [11]. Neural ODEs (formulated with respect to time) are useful for constructing continuous-time trajectories and have been explored before to model single-cell development [42,49,13].

`scNODE` uses neural ODEs to parameterize the continuous dynamics of gene expression in the latent space. Specifically, `scNODE` quantifies changes of cell latent representation $\mathbf{Z}^{(t)}$ (from VAE encoding) at time t through a neural ODE

$$\mathrm{d}\mathbf{Z}^{(t)} = \text{Drift}_\omega\left(\mathbf{Z}^{(t)}; t\right) \cdot \mathrm{d}t \tag{3}$$

Here, Drift_ω is a non-linear neural network with parameterization ω, modeling the developmental velocities in the latent space, such that $\text{Drift}_\omega(\mathbf{Z}^{(t)}; t)$ represents the direction and strength of cellular transitions. `scNODE` computes the latent space $\mathbf{Z}^{(0)}$ of cells at $t = 0$ through the encoder network (pre-trained in the previous step) and predicts the subsequent cell states step-wise at any timepoint t as follows:

$$\begin{gathered}\mathbf{Z}^{(0)} \sim \mathcal{N}(\mu_0, \sigma_0) \quad \text{and} \quad \mu_0, \sigma_0 = \text{Enc}_\phi(\mathbf{X}^{(0)}), \\ \mathbf{Z}^{(t+\Delta t)} = \mathbf{Z}^{(t)} + \text{Drift}_\omega\left(\mathbf{Z}^{(t)}, t\right) \Delta t.\end{gathered} \tag{4}$$

Here, hyperparameter Δt denotes step size and $\text{Drift}_\omega\left(\mathbf{Z}^{(t)}, t\right)\Delta t$ represents the forward steps taken in the latent space. While there are several methods that can solve this ODE, in this work, we apply the commonly used first-order Euler method (in Eq. 4) for convenience of explanation. However, in our implementation, one can specify any ODE solver.

To fit the continuous trajectory (controlled by Drift_ω) to the observations, `scNODE` minimizes the difference between the input and the reconstructed gene expression. Specifically, at each measured timepoint $t \in \mathcal{T}$, `scNODE` uses the decoder Dec_θ to convert latent variables $\mathbf{Z}^{(t)}$ generated from Eq. 4 back to the gene space through $\widehat{\mathbf{X}}^{(t)} = \text{Dec}_\theta\left(\mathbf{Z}^{(t)}\right)$. Because we have no correspondence between true cells and cells generated from the ODE model, `scNODE` utilizes the Wasserstein metric [15] to measure the distance between distributions defined by ground truth $\mathbf{X}$ and predictions $\widehat{\mathbf{X}}$ as

$$\text{Wass}(\mathbf{X}, \widehat{\mathbf{X}}) = \left(\min_{\Gamma \sim \Pi(\mathbf{X}, \widehat{\mathbf{X}})} \sum_{i,j} D_{ij}^2 \Gamma_{ij}\right)^{1/2} \quad \text{with} \quad D_{ij} = \|\mathbf{X}_i - \widehat{\mathbf{X}}_j\|_2. \tag{5}$$

Here, $\Pi(\mathbf{X}, \widehat{\mathbf{X}})$ denotes the set of all transport plans between each cell of $\mathbf{X}$ and $\widehat{\mathbf{X}}$ and D_{ij} represents the ℓ_2 norm distance, such that the Wasserstein metric adopts the minimal-cost transport plan Γ to measure the data dissimilarity. Furthermore, `scNODE` should be adjusted to ensure that the latent variables generated by the ODE (i.e., $\mathbf{Z}$) do not shift from the global structure learned by the VAE (i.e., $\text{Enc}_\phi(\mathbf{X})$) through the Wasserstein distance between them. Therefore, `scNODE` optimizes parameters ϕ, θ, and ω by minimizing regularized loss function

$$\begin{gathered}\mathcal{L}_{\text{dynamic}} = \sum_{t \in \mathcal{T}} \text{Wass}\left(\mathbf{X}^{(t)},\ \widehat{\mathbf{X}}^{(t)}\right) \ + \ \beta \sum_{t \in \mathcal{T}} \text{Wass}\left(\mathbf{Z}_{\text{enc}}^{(t)},\ \mathbf{Z}^{(t)}\right) \\ \text{with} \quad \mathbf{Z}_{\text{enc}}^{(t)} \sim \mathcal{N}(\mu_t, \sigma_t) \quad \text{and} \quad \mu_t, \sigma_t = \text{Enc}_\phi(\mathbf{X}^{(t)}).\end{gathered} \tag{6}$$

The final latent space of `scNODE` is fine-tuned further by the overall dynamics in the data as regularized by the hyperparameter β. Specifically, at each observed timepoint $t \in \mathcal{T}$, we use the encoder to compute latent variables $\mathbf{Z}_{\text{enc}}^{(t)}$ for $\mathbf{X}^{(t)}$ and then compute the Wasserstein distance between it and latent variables $\mathbf{Z}^{(t)}$ from ODE (Eq. 4).

This regularization term is a key difference from previous generative models [52,55,28,68], which learn cell dynamics only in the fixed PCA latent space. The PCA space is computed from all measured timepoints and fixed during modeling dynamics, which cannot account for cell variabilities and developments at unmeasured timepoints. This may impose biases or errors in gene expression prediction, especially for timepoints beyond the measured time range. On the contrary, `scNODE` fine-tunes the latent space by posing cellular dynamics to the latent space and combining structure relation and temporal development of cells, which leads to an informative and interpretable latent space. This improves the generalizability of `scNODE` in predicting accurate gene expression of cells for unobserved timepoints and helps downstream analysis based on the latent representations. Algorithm 1 gives the pseudo-code for `scNODE` .

During `scNODE` training, we use Eq. 4 to predict cell latent representations at observed timepoints and convert them back to gene expression space through the VAE decoder. During inference, `scNODE` can use Eq. 4 to predict gene expression at unmeasured timepoints and utilize these predictions for downstream analysis.

Algorithm 1 `scNODE`

1: **Input:** The set of measured timepoint indices $\mathcal{T}$; expression matrices $\{\mathbf{X}^{(t)} \mid t \in \mathcal{T}\}$; hyperparameters Δt, λ, and β; randomly initialized neural networks Enc_ϕ, Dec_θ, and Drift_ω.
2: // Pre-Training
3: $\mathbf{X}_{\text{ALL}} = \text{CONCAT}\left(\mathbf{X}^{(t)} \mid t \in \mathcal{T}\right)$ // concatenate cells from training timepoints
4: Optimize ϕ and θ to minimize $\mathcal{L}_{\text{pre-train}}$ (Eq. 2)
5: // Dynamic Modelling
6: Optimize ϕ, θ, and ω to minimize $\mathcal{L}_{\text{dynamic}}$ (Eq. 6)
7: **Output:** ϕ, θ, and ω

3 Results

3.1 Experimental Setup

Single-cell datasets We use six publicly available scRNA-seq datasets to demonstrate the capabilities of `scNODE` in predicting developmental dynamics from real-world transcriptomic data. These datasets are summarized in Table 1 and cover various species, tissues, and the numbers of recorded timepoints. In each experiment, we select the top 2000 most highly variable genes (HVGs) from the datasets and normalize the unique molecular identifier (UMI) count expression through a log transformation with pseudo-count.

Table 1. Data descriptions of the six real-world scRNA-seq datasets used in experiments.

ID	Dataset	Specie	# Cells	# Timepoints	Source
EB	embryoid body	Homo sapiens	31161	5	[45]
MP	mouse pancreatic	Mus musculus	37932	4	GSE132188 [3]
ZB	zebrafish embryo	Danio rerio	38731	12	Broad Single-Cell Portal SCP162 [21]
MB	mouse cortex	Mus musculus	98047	13	Broad Single-Cell Portal SCP1290 [17]
DR	drosophila	Drosophila	27386	11	GSE190149 [8]
SC	Schiebinger2019	Mus musculus	236285	39	[52]

Training and testing We test `scNODE` performance in predicting gene expression at missing timepoints. Specifically, for each dataset, we remove several timepoints in the middle of the measured time range (for the **interpolation** task) or the last few timepoints (for the **extrapolation** task) to test whether `scNODE` can recover these left-out observations. Table 2 shows the timepoints removed in each dataset. Because EB and MP datasets have no more than 5 timepoints, we remove a single time point for both tasks. For ZB, MB, DR, and SC data, which have a greater number of measured timepoints, we have left multiple timepoints out. We consider left-out timepoints as the testing set and the remaining timepoints as the training set. In each case, HVGs are selected based on cells corresponding to the training timepoints in order to avoid data leakage.

`scNODE` is implemented with *Pytorch 1.13* [47] and is trained end-to-end. `scNODE` training consists of two main steps (Algorithm 1). `scNODE` first pre-trains the VAE component (encoder Enc_ϕ and decoder Dec_θ) with cells corresponding to the training timepoints. We use Adam optimizer to pre-train `scNODE` with a learning rate of 0.001 and 200 iterations. Then, `scNODE` optimizes both VAE and neural ODE components by minimizing the $\mathcal{L}_{\text{dynamic}}$. We adopt batch training and use Adam optimizer to train `scNODE` with a learning rate of 0.001 and 1000 iterations. Specifically, at each training iteration, we select 32 cells at $t = 0$ as a batch and predict gene expression at every $t \in \mathcal{T}$. Because the Wasserstein distance computation is expensive, batch training improves training efficiency and enables `scNODE` usage on large-scale datasets. We use *geomloss* [22] to compute Wasserstein distance with blur=0.05 and scaling=0.5.

At inference, `scNODE` first uses the pre-trained encoder Enc_ϕ to map gene expression at the first timepoint to the latent space and then samples latent variables for the required number of cells from the parameterized Gaussian distribution. Then, `scNODE` uses the ODE solver (Eq. 4) to predict intermediate or future states of the latent variables of these cells at intervals of Δt and maps them back to the gene space through the decoder Dec_θ. We set step size $\Delta t = 0.1$ for `scNODE` training and inference.

Table 2. The leave-out timepoints (testing timepoints) for the interpolating and extrapolating tasks in each dataset. The timepoint index starts with 0.

Dataset	# Timepoints	Leave-Out Timepoints (Testing Timepoints)	
		Interpolating	Extrapolating
EB	5	2	4
MP	4	2	3
ZB	12	4, 6, 8	10, 11
MB	13	4, 6, 8	10, 11, 12
DR	11	4, 6, 8	8, 9, 10
SC	39	15, 20, 25	36, 37, 38

Baselines We compare `scNODE` with three state-of-the-art generative methods.

- **PRESCIENT**: Yeo et al. [68] propose a generative model, called Potential eneRgy undErlying Single Cell gradIENTs (`PRESCIENT`), to learn the differentiation landscape from single-cell time-series gene expression data. `PRESCIENT` maps gene expression to PCA space and models cellular differentiation with neural ODE.
- **WOT**: Waddintong-OT (`WOT`) [52] is an approach for studying time-series scRNA-seq data to infer ancestor-descendant fates. `WOT` can recover a held-out timepoint through linear interpolation based on transport maps constructed between sets of cells in early and late timepoints.
- **TrajectoryNet**: Tong et al. [55] utilize continuous normalizing flows [11] to model the paths of cells over time and interpolate between measured time points. `TrajectoryNet` maps gene expression to PCA space and models cellular dynamics by solving a dynamic OT problem [5]

Moreover, we add an intuitive baseline method, `Naive` model, which makes predictions by replicating expressions of the closest timepoint.

- **Naive**: The `Naive` model replicates the true gene expression of the last observed timepoint for prediction. For example, if we predict $t = 3, 4$ from $t = 0, 1, 2$, `Naive` takes $\mathbf{X}^{(2)}$ as the prediction at timepoints $t = 3$ and $t = 4$. The `Naive` model gives computationally costless prediction without learning.

Because `TrajecroyNet` and `WOT` can only do interpolation, we only compare `scNODE` with `PRESCIENT` and `Naive` for the extrapolation task. Moreover, `TrajectoryNet` can only predict one timepoint at a time. Hence, we exclude it for the ZB, MB, DR, and SC experiments, where we have multiple timepoints that are all left out.

Evaluation metrics We use the Wasserstein distance and average pair-wise ℓ_2 distance to measure the similarity between true and predicted expression at each testing timepoint. These evaluation metrics have been used in previous studies [68,55]. Specifically, given the true expression $\mathbf{X}^{(t)} \in \mathbb{R}^{n_1 \times p}$ and predicted expression $\widehat{\mathbf{X}}^{(t)} \in \mathbb{R}^{n_2 \times p}$, we compute the Wasserstein distance with the Sinkhorn algorithm [15] and compute the pair-wise ℓ_2 distance through

$$\ell_2\left(\mathbf{X}^{(t)}, \widehat{\mathbf{X}}^{(t)}\right) = \frac{1}{n_1 n_2} \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \|\mathbf{X}_i^{(t)} - \widehat{\mathbf{X}}_j^{(t)}\|_2. \tag{7}$$

We compute the metric for each predicted timepoint and report the average value. For both metrics, a lower value corresponds to more accurate predictions.

Hyperparameter tuning We select hyperparameters that yield the minimum averaged Wasserstein distance for `scNODE` and `PRESCIENT` through 3-fold cross-validation. For example, if the dataset has timepoints $t = 0, 1$ and we want to predict timepoint $t = 2$, then $\mathbf{X}^{(2)}$ is the testing set. In cross-validation, we equally split the cells at every timepoint of $t = 0, 1$ into three subsets, train the model on two of them, and validate it on the remainder. We repeat this process three times to ensure each subset is used as the validation set once and record the averaged Wasserstein distance on three validation sets. We use Optuna [1] to automatically determine the optimal hyperparameters and use sufficiently large hyperparameter ranges for search and evaluation.

For `WOT` and `TrajectoryNet` that can only predict interpolations between two timepoints, it is difficult to define the validation set, so cross-validation is infeasible. For instance, assume we use timepoints $t = 0, 2$ to predict timepoint $t = 1$. The validation set is undefined because it is supposed to be expression data at training time points $t = 0, 2$, but these two models cannot predict data at $t = 0$ or 2, such that it is impossible to evaluate the model on validation sets. Therefore, we use grid search to find the best hyperparameter setting with the minimum averaged Wasserstein distance on testing timepoints for `WOT` and `TrajectoryNet`. We define a large grid for `WOT` and `TrajectoryNet` to ensure a wide range of hyperparameter settings are evaluated. The hyperparameter search space for all methods is listed in Appendix Table A1.

We set the latent space dimension as 50 for all methods and run each method for sufficient iterations to ensure they converge. We follow the `PRESCIENT` setting because it is the main baseline model. We let every

[5] `MIOFlow` [28] implementation has the same functionality as `TrajectoryNet`, interpolating one timepoint at a time. But it performs worse than `TrajectoryNet` in our preliminary experiments. Therefore, we choose `TrajectoryNet` as a representative.

model predict 2000 cells at each testing timepoint to ensure a fair metric comparison. We find the number of predicted cells has little to no effect on evaluation metrics. Moreover, datasets used in our experiments have thousands of cells at each timepoint, so we adopt 2000 as a reasonable number.

3.2 `scNODE` can accurately predict expression at unmeasured timepoints

We first compare `scNODE` 's performance with baseline methods for predicting gene expression for left-out timepoints. Fig. 2 shows the Wasserstein distance and pair-wise ℓ_2 distance averaged over the left-out testing timepoints. `scNODE` clearly outperforms the baselines in extrapolation tasks for all six datasets and interpolation tasks for five out of six datasets. For the Wasserstein distance for the MP interpolation task, `TrajectoryNet` has a slightly better performance than `scNODE` . However, for the other three comparisons that can be made between the two methods - MP ℓ_2 distance, EB ℓ_2 distance, and EB Wasserstein distance - `scNODE` performs better than `TrajectoryNet`. Furthermore, `TrajectoryNet` is only limited to single time-point interpolations and cannot predict multiple consecutive missing timepoints. Therefore, `scNODE` has better overall performance. Appendix Sec. E provides detailed evaluation metric values of each model on every dataset.

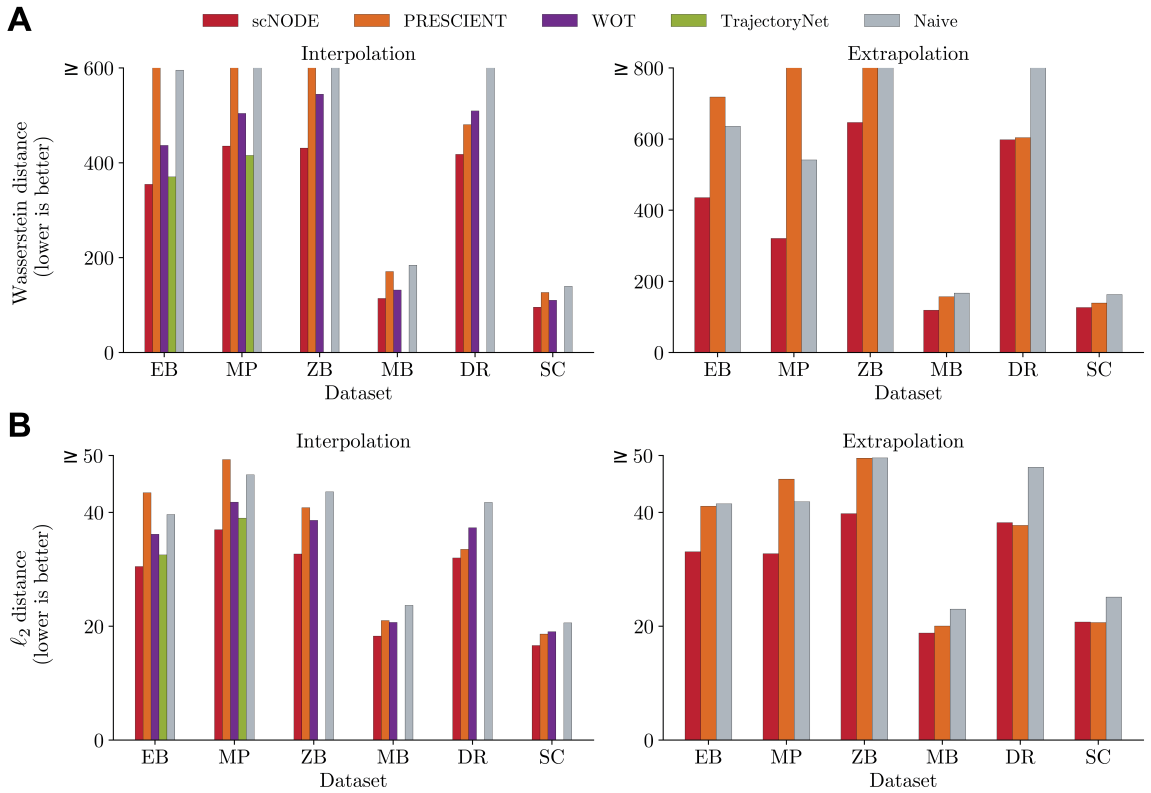

Fig. 2. `scNODE` prediction results. Averaged Wasserstein distance (**A**) and pair-wised ℓ_2 distance (**B**) between true and predicted expression on leave-out timepoints.

Fig. 3 and Appendix Fig. A6 and Fig. A7 visualize true gene expression values and model predictions in the 2D Uniform Manifold Approximation and Projection (UMAP) [44] space. Our results indicate `scNODE` 's predictions align well with ground truth. Qualitative evaluation of six scRNA-seq datasets indicates that `scNODE` can accurately predict gene expression at missing timepoints.

Unsurprisingly, extrapolating into the future is more difficult than interpolating because the training data does not contain information about future time points. The extrapolations become less accurate the farther out predictions are made. Appendix Fig. A1 shows that `scNODE` prediction accuracy decreases as a function of time. We assess the performance deterioration on three datasets (ZB, DR, and MB) with many timepoint measurements and notice the deterioration exists in different species or organs but has different rates. For example, the rate at which the performance declines is low on the MB dataset such that

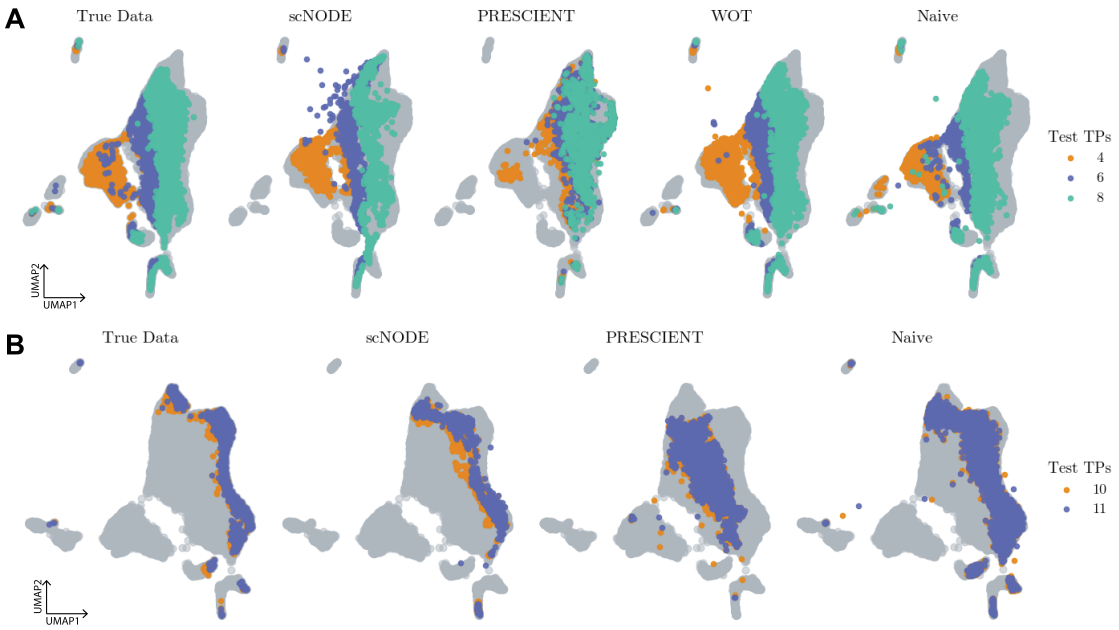

Fig. 3. 2D UMAP visualization of true and predicted expression of interpolation (**A**) and extrapolation (**B**) tasks in ZB dataset.

extrapolations to the next 4 to 5 timepoints have similar scores, while the rate at which the performance declines is high in DR dataset, where the predictive accuracy suddenly decreases. In addition, we show the 2D UMAP of extrapolating predictions (Appendix Fig. A2) and find that the farther the model predicts into the future, the further the qualitative agreement degrades. Although accurate extrapolation is a challenge [63] in single-cell data prediction, `scNODE` still demonstrates significant improvement over state-of-the-art methods in accurately extrapolating proximal timepoints (Fig. 2).

3.3 `scNODE` assists with perturbation analysis of gene expression

The ability of `scNODE` to learn informative latent space with integrated dynamics and generate trajectories allows it to detect key driver genes for developmental paths and simulate cells with perturbed gene expression, which is important for *in silico* perturbation predictions. We validate this ability with the differential dynamics learned from ZB dataset. We choose ZB data because the original dataset provides cell type labels at the last timepoint, which enables us to validate perturbation predictions.

Construct cell transition path We train `scNODE` with all timepoints in ZB dataset and map all cells to the latent space with the learned `scNODE` encoder. Since the encoder's weights are tuned with the ODE component, the VAE latent space effectively represents the overall cell relationships and developments. In the latent space, we construct a most probable path between any two points through the Least Action Path (LAP) method [49,48,61]. LAP finds the optimal path between two cell states while minimizing its action and transition time (see Appendix Section C). The optimal path is not simply the algebraically shortest path in the gene expression space but follows the cell differential landscape in latent space modeled by `scNODE` . Here, we construct LAP paths from cells at the starting point (i.e., $t = 0$) to presomitic mesoderm (PSM) and Hindbrain cell populations (Fig. 4**A**)

Detect key genes Once the optimal path is constructed, we can associate gene expression with points along the path and detect key driver genes controlling the developmental processes. Specifically, for each node on the path, we project it back to the gene expression space and find the nearest neighbors. The differentially expressed (DE) genes of these neighbor expression profiles represent key driver genes for the developmental trajectories. We use the Wilcoxon rank-sum test to find DE genes for each path. Fig. 4**B** and Appendix Fig. A8**B** show the expression of two DE genes (TBX16, TOB1A) for the PSM path and two DE genes (SOX3, SOX19A) for the Hindbrain path, such that their respective expression levels vary along the path (as well as in adjacent regions). Xiong et al. [67] have found that TOB1A is required for correct dorsoventral patterning and Warga et al. [62] have found TBX16 regulates intermediate mesoderm cell fate. Other studies [16,59] have found SOX3 and SOX19A regulate neural fates and are involved in central nervous system development.

Simulate with perturbed gene expression Finally, we perturb the expression profiles of key genes in all cells at the starting timepoint and let `scNODE` predict trajectories for the perturbed gene expression. The perturbations are expected to result in changes in cell fates. Specifically, we overexpress TBX16 (DE gene associated with the PSM path) or SOX3 (DE gene associated with the Hindbrain path) by multiplying their expression values with 100 at $t = 0$ and using `scNODE` to predict cells at the last timepoint. We classify predicted cells with a Random Forest (RF) classifier [7] trained with all unperturbed cells and find the ratio of PSM cells increases from 12.26% (unperturbed) to 52% (TBX16 perturbed) and the ratio of Hindbrain cells increases from 11% (unperturbed) to 22% (SOX3 perturbed) after perturbation (Fig. 4**C** and Appendix Fig. A8**C**), indicating expected cell fate outcomes. In applications of `scNODE` , these perturbations can involve multiple genes, use different cell populations, or be carried out at different timepoints.

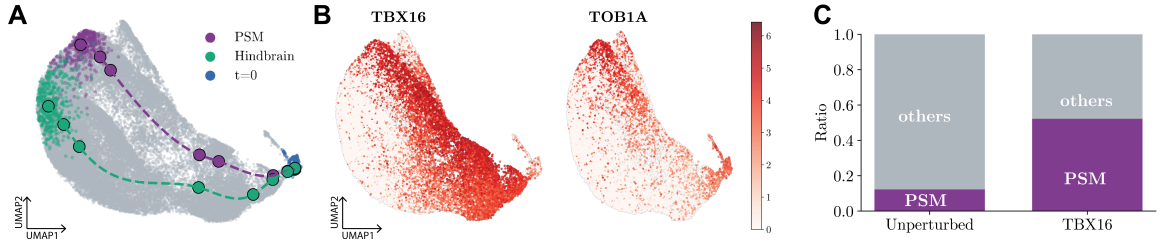

Fig. 4. `scNODE` perturbation analysis results. (**A**) 2D UMAP visualization of the least action path between cells at the starting point ($t = 0$) and the PSM and Hindbrain cell populations, respectively. (**B**) Gene expression value of two DE genes (TBX16 and TOB1A) associated with the PSM path. (**C**) The ratio of PSM cells and other cell types in unperturbed data and predictions from TBX16 perturbations.

3.4 `scNODE` predictions help recover cell trajectories

To determine if `scNODE` can aid with temporal downstream analysis, we carry out trajectory analysis on both real and predicted single-cell gene expression. Specifically, we remove cells at intermediate timepoints $t = 2, 4, 6, 8$ and last timepoints $t = 10, 11$ from the ZB dataset and use `scNODE` to simulate them. Fig. 5 visualizes the ZB data with all timepoints, after the removal of six timepoints, and with predictions from `scNODE` . It can be seen that after the removal of the intermediate timepoints, the cellular landscape has gaps between consecutive discrete timepoints, which can impede the trajectory analysis where one would expect to learn smooth trajectories. But `scNODE` can fill gaps by predicting accurate interpolations. We also apply partition-based graph abstraction (PAGA) [65], a method that computes the topological structures of cellular populations and has been used in many studies [51,40,25] to understand cell developmental topology and to construct cell trajectories. We find that after timepoint removal, the topology breaks down due to the gaps between timepoints, while `scNODE` predictions recover smooth and continuous trajectories. We use the Ipsen-Mikhailov (IM) distance [30] to quantitatively measure the similarity between the cell trajectory graphs constructed in each case. $\mathrm{IM}(\mathcal{G}_1, \mathcal{G}_2)$ is a graph similarity measurement defined as the square-root difference between the Laplacian spectrum of graphs $\mathcal{G}_1$ and $\mathcal{G}_2$. It ranges from 0 to 1, where 0 indicates maximum similarity between two graph structures and 1 indicates maximum dissimilarity. Specifically, we let $\mathcal{G}_{\text{true}}$, $\mathcal{G}_{\text{removal}}$, and $\mathcal{G}_{\text{pred}}$ denote the PAGA topological structures constructed from all true cells, cells after timepoint removal, and cells at missing timepoints augmented with `scNODE` predictions, respectively. We find $\mathrm{IM}(\mathcal{G}_{\text{true}}, \mathcal{G}_{\text{removal}}) = 0.202$ and $\mathrm{IM}(\mathcal{G}_{\text{true}}, \mathcal{G}_{\text{pred}}) = 0.093$, indicating $\mathcal{G}_{\text{pred}}$ is more similar to $\mathcal{G}_{\text{true}}$ than $\mathcal{G}_{\text{removal}}$ such that `scNODE` predictions help recovering cell trajectories.

3.5 Investigation of relevant hyperparameters and pre-training step for `scNODE`

We evaluate `scNODE` performance using different hyperparameter settings and give heuristic guidance on how to set hyperparameters in real-world scenarios. All experiments are conducted for the interpolation task of the ZB dataset. First, we vary the latent dimension from $\{25, 50, 75, \cdots, 200\}$. Appendix Table A8 indicates that `scNODE` is robust to the size of the latent dimensionality. However, latent size can affect computational cost. Users can choose to set a reasonable latent dimension based on a tradeoff between accuracy and computational costs. State-of-the-art methods [56,57,27] generally choose a latent space of 10 to 50 dimensions. Moreover, `scNODE` uses a regularizer controlled by β to interpolate between VAE latent space and the ODE manifold. The VAE latent space retains the global structure of cell-cell relationships but can lead to inaccurate predictions for missing timepoints. On the contrary, the ODE manifold is trained to improve predictive performance by considering overall dynamics but may not preserve the relationships

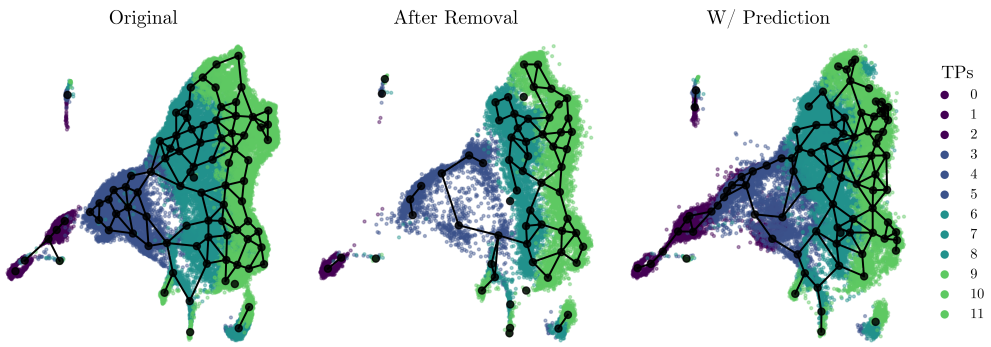

Fig. 5. Cell trajectories of ZB data with all timepoints (*left*), after removal of timepoints $t = 2, 4, 6, 8, 10, 11$ (*middle*), and with `scNODE` predictions (*right*). The connective structure is constructed with PAGA where black nodes represent cell clusters and edges connect two nodes if their expressions are similar in the low-dimensional space.

among cells. Therefore, the regularizer coefficient β sets the trade-off between accuracy and interpretation. We vary $\beta \in \{0.0, 0.25, 0.5, 0.75, 1.0\}$ and Appendix Fig. A3 shows that a larger β leads to better estimations but less-interpretable latent space such that different differential trajectories of distinct cell types are less distinguishable. We set $\beta = 1$ for `scNODE` in all experiments and find it is a reasonable setting to obtain both generalizability and predictive performance.

Lastly, we conduct an ablation study to test the effects of the pre-training phase. Specifically, we compare the accuracy of `scNODE` predictions when including or excluding the pre-training step. Appendix Fig. A4 shows that pre-training improves predictions for some datasets (MP, ZB, and DR), while it slightly worsens predictions in other cases. However, regardless of the inclusion of the pre-training step, `scNODE` still outperforms baseline models. In addition, we compared the latent space learned from `scNODE` with and without pre-training for a qualitative assessment (Appendix Fig. A5). Results indicate the improvement in interpretation is data-dependent. Specifically, on ZB data, the latent space from `scNODE` without pre-training better separates different cell types, but on the MB dataset the pre-training helps unmix heterogenous cell groups. Overall, the pre-training has no significant negative effects on prediction accuracies and has the potential to improve latent space interpretation.

4 Discussion and Conclusion

In this paper, we propose a generative model called `scNODE` that can be used to simulate the changes in the gene expression of cellular populations over time. Given a time-series scRNA-seq data set, our model can both interpolate between timepoints or extrapolate beyond them. Filling in these gaps enables many types of downstream analyses, such as differential gene expression analysis, perturbation analysis, and trajectory inference, among others. To make these predictions, `scNODE` uses a non-linear dimensionality reduction approach and updates the latent space during dynamic learning. This improves the interpretability and generalizability of the learned latent representation. We validate `scNODE` on six real-world scRNA-seq datasets and show that `scNODE` consistently outperforms state-of-the-art methods. We construct the least action transition path between cell states, detect key driver genes of the path, and conduct *in silico* perturbation analysis in the latent space learned by `scNODE` . We also use `scNODE` to help recapitulate smooth cellular trajectories by predicting unmeasured timepoints. They have shown the application of `scNODE` in studying differential processes that take place over continuous time.

We find the primary challenge for the scRNA-seq prediction task is extrapolating to multiple timepoints beyond the most proximal timepoint, as the accuracy deteriorates the farther ahead into the future predictions are made. This is reasonable since measured timepoints contain limited information about future development. We suggest that in follow-up work, knowledge about cell proliferation - which is known to be important to cellular differentiation - should be incorporated into the model to improve performance. Moreover, although RNA velocity methods have inevitable limitations on gene expression prediction, they can perform as complementary approaches of `scNODE` for understanding cellular dynamics.

In summary, we envision that `scNODE` will be helpful for various types of temporal single-cell analyses, especially for diseased cells such as tumor cells or stem cell research where samples are difficult to profile. Our model provides a valuable tool for the community as we work towards understanding complex biological processes from temporally recorded single-cell data.

References

1. Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: A next-generation hyperparameter optimization framework. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining. pp. 2623–2631 (2019)
2. Ball, T.M., Squeglia, L.M., Tapert, S.F., Paulus, M.P.: Double dipping in machine learning: problems and solutions. Biological psychiatry. Cognitive neuroscience and neuroimaging **5**(3), 261 (2020)
3. Bastidas-Ponce, A., Tritschler, S., Dony, L., Scheibner, K., Tarquis-Medina, M., Salinno, C., Schirge, S., Burtscher, I., Böttcher, A., Theis, F.J., et al.: Comprehensive single cell mrna profiling reveals a detailed roadmap for pancreatic endocrinogenesis. Development **146**(12), dev173849 (2019)
4. Bergen, V., Lange, M., Peidli, S., Wolf, F.A., Theis, F.J.: Generalizing rna velocity to transient cell states through dynamical modeling. Nature biotechnology **38**(12), 1408–1414 (2020)
5. Bergen, V., Soldatov, R.A., Kharchenko, P.V., Theis, F.J.: Rna velocity—current challenges and future perspectives. Molecular systems biology **17**(8), e10282 (2021)
6. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. Journal of statistical mechanics: theory and experiment **2008**(10), P10008 (2008)
7. Breiman, L.: Random forests. Machine learning **45**, 5–32 (2001)
8. Calderon, D., Blecher-Gonen, R., Huang, X., Secchia, S., Kentro, J., Daza, R.M., Martin, B., Dulja, A., Schaub, C., Trapnell, C., et al.: The continuum of drosophila embryonic development at single-cell resolution. Science **377**(6606), eabn5800 (2022)
9. Cao, Y., Yang, P., Yang, J.Y.H.: A benchmark study of simulation methods for single-cell rna sequencing data. Nature communications **12**(1), 6911 (2021)
10. Chen, G., Ning, B., Shi, T.: Single-cell rna-seq technologies and related computational data analysis. Frontiers in genetics p. 317 (2019)
11. Chen, R.T., Rubanova, Y., Bettencourt, J., Duvenaud, D.K.: Neural ordinary differential equations. Advances in neural information processing systems **31** (2018)
12. Chen, W., Guillaume-Gentil, O., Rainer, P.Y., Gäbelein, C.G., Saelens, W., Gardeux, V., Klaeger, A., Dainese, R., Zachara, M., Zambelli, T., et al.: Live-seq enables temporal transcriptomic recording of single cells. Nature **608**(7924), 733–740 (2022)
13. Chen, Z., King, W.C., Hwang, A., Gerstein, M., Zhang, J.: Deepvelo: Single-cell transcriptomic deep velocity field learning with neural ordinary differential equations. Science Advances **8**(48), eabq3745 (2022)
14. Csiszár, I.: I-divergence geometry of probability distributions and minimization problems. The annals of probability pp. 146–158 (1975)
15. Cuturi, M.: Sinkhorn distances: Lightspeed computation of optimal transport. Advances in neural information processing systems **26** (2013)
16. Dee, C.T., Hirst, C.S., Shih, Y.H., Tripathi, V.B., Patient, R.K., Scotting, P.J.: Sox3 regulates both neural fate and differentiation in the zebrafish ectoderm. Developmental Biology **320**(1), 289–301 (2008)
17. Di Bella, D.J., Habibi, E., Stickels, R.R., Scalia, G., Brown, J., Yadollahpour, P., Yang, S.M., Abbate, C., Biancalani, T., Macosko, E.Z., et al.: Molecular logic of cellular diversification in the mouse cerebral cortex. Nature **595**(7868), 554–559 (2021)
18. Di Micco, R., Krizhanovsky, V., Baker, D., d'Adda di Fagagna, F.: Cellular senescence in ageing: from mechanisms to therapeutic opportunities. Nature reviews Molecular cell biology **22**(2), 75–95 (2021)
19. Ding, J., Condon, A., Shah, S.P.: Interpretable dimensionality reduction of single cell transcriptome data with deep generative models. Nature communications **9**(1), 2002 (2018)
20. Ding, J., Sharon, N., Bar-Joseph, Z.: Temporal modelling using single-cell transcriptomics. Nature Reviews Genetics **23**(6), 355–368 (2022)
21. Farrell, J.A., Wang, Y., Riesenfeld, S.J., Shekhar, K., Regev, A., Schier, A.F.: Single-cell reconstruction of developmental trajectories during zebrafish embryogenesis. Science **360**(6392), eaar3131 (2018)
22. Feydy, J., Séjourné, T., Vialard, F.X., Amari, S.i., Trouvé, A., Peyré, G.: Interpolating between optimal transport and mmd using sinkhorn divergences. In: The 22nd International Conference on Artificial Intelligence and Statistics. pp. 2681–2690. PMLR (2019)
23. Fleck, J.S., Jansen, S.M.J., Wollny, D., Zenk, F., Seimiya, M., Jain, A., Okamoto, R., Santel, M., He, Z., Camp, J.G., et al.: Inferring and perturbing cell fate regulomes in human brain organoids. Nature pp. 1–8 (2022)
24. Grønbech, C.H., Vording, M.F., Timshel, P.N., Sønderby, C.K., Pers, T.H., Winther, O.: scvae: variational auto-encoders for single-cell gene expression data. Bioinformatics **36**(16), 4415–4422 (2020)
25. Han, X., Zhou, Z., Fei, L., Sun, H., Wang, R., Chen, Y., Chen, H., Wang, J., Tang, H., Ge, W., et al.: Construction of a human cell landscape at single-cell level. Nature **581**(7808), 303–309 (2020)
26. Hao, Y., Hao, S., Andersen-Nissen, E., Mauck, W.M., Zheng, S., Butler, A., Lee, M.J., Wilk, A.J., Darby, C., Zager, M., et al.: Integrated analysis of multimodal single-cell data. Cell **184**(13), 3573–3587 (2021)

27. Heumos, L., Schaar, A.C., Lance, C., Litinetskaya, A., Drost, F., Zappia, L., Lücken, M.D., Strobl, D.C., Henao, J., Curion, F., et al.: Best practices for single-cell analysis across modalities. Nature Reviews Genetics pp. 1–23 (2023)
28. Huguet, G., Magruder, D.S., Tong, A., Fasina, O., Kuchroo, M., Wolf, G., Krishnaswamy, S.: Manifold interpolating optimal-transport flows for trajectory inference. Advances in Neural Information Processing Systems **35**, 29705–29718 (2022)
29. Hwang, B., Lee, J.H., Bang, D.: Single-cell rna sequencing technologies and bioinformatics pipelines. Experimental & molecular medicine **50**(8), 1–14 (2018)
30. Ipsen, M.: Evolutionary reconstruction of networks. Function and regulation of cellular systems pp. 241–249 (2004)
31. Kidger, P.: On neural differential equations. arXiv preprint arXiv:2202.02435 (2022)
32. Kingma, D.P., Welling, M., et al.: An introduction to variational autoencoders. Foundations and Trends® in Machine Learning **12**(4), 307–392 (2019)
33. Kobak, D., Berens, P.: The art of using t-sne for single-cell transcriptomics. Nature communications **10**(1), 5416 (2019)
34. Kriegeskorte, N., Simmons, W.K., Bellgowan, P.S., Baker, C.I.: Circular analysis in systems neuroscience: the dangers of double dipping. Nature neuroscience **12**(5), 535–540 (2009)
35. La Manno, G., Soldatov, R., Zeisel, A., Braun, E., Hochgerner, H., Petukhov, V., Lidschreiber, K., Kastriti, M.E., Lönnerberg, P., Furlan, A., et al.: Rna velocity of single cells. Nature **560**(7719), 494–498 (2018)
36. Lähnemann, D., Köster, J., Szczurek, E., McCarthy, D.J., Hicks, S.C., Robinson, M.D., Vallejos, C.A., Campbell, K.R., Beerenwinkel, N., Mahfouz, A., et al.: Eleven grand challenges in single-cell data science. Genome biology **21**(1), 1–35 (2020)
37. Li, W.V., Li, J.J.: A statistical simulator scdesign for rational scrna-seq experimental design. Bioinformatics **35**(14), i41–i50 (2019)
38. Lin, E., Mukherjee, S., Kannan, S.: A deep adversarial variational autoencoder model for dimensionality reduction in single-cell rna sequencing analysis. BMC bioinformatics **21**(1), 1–11 (2020)
39. Liu, B., Hu, X., Feng, K., Gao, R., Xue, Z., Zhang, S., Zhang, Y., Corse, E., Hu, Y., Han, W., et al.: Temporal single-cell tracing reveals clonal revival and expansion of precursor exhausted t cells during anti-pd-1 therapy in lung cancer. Nature Cancer **3**(1), 108–121 (2022)
40. Luecken, M.D., Theis, F.J.: Current best practices in single-cell rna-seq analysis: a tutorial. Molecular systems biology **15**(6), e8746 (2019)
41. Marouf, M., Machart, P., Bansal, V., Kilian, C., Magruder, D.S., Krebs, C.F., Bonn, S.: Realistic in silico generation and augmentation of single-cell rna-seq data using generative adversarial networks. Nature communications **11**(1), 166 (2020)
42. Matsumoto, H., Kiryu, H., Furusawa, C., Ko, M.S., Ko, S.B., Gouda, N., Hayashi, T., Nikaido, I.: Scode: an efficient regulatory network inference algorithm from single-cell rna-seq during differentiation. Bioinformatics **33**(15), 2314–2321 (2017)
43. McCabe, S., Torres, L., LaRock, T., Haque, S.A., Yang, C.H., Hartle, H., Klein, B.: netrd: A library for network reconstruction and graph distances. arXiv preprint arXiv:2010.16019 (2020)
44. McInnes, L., Healy, J., Melville, J.: Umap: Uniform manifold approximation and projection for dimension reduction. arXiv preprint arXiv:1802.03426 (2018)
45. Moon, K.R., van Dijk, D., Wang, Z., Gigante, S., Burkhardt, D.B., Chen, W.S., Yim, K., Elzen, A.v.d., Hirn, M.J., Coifman, R.R., et al.: Visualizing structure and transitions in high-dimensional biological data. Nature biotechnology **37**(12), 1482–1492 (2019)
46. Moris, N., Pina, C., Arias, A.M.: Transition states and cell fate decisions in epigenetic landscapes. Nature Reviews Genetics **17**(11), 693–703 (2016)
47. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems **32** (2019)
48. Qiu, X., Ding, S., Shi, T.: From understanding the development landscape of the canonical fate-switch pair to constructing a dynamic landscape for two-step neural differentiation. PloS one **7**(12), e49271 (2012)
49. Qiu, X., Zhang, Y., Martin-Rufino, J.D., Weng, C., Hosseinzadeh, S., Yang, D., Pogson, A.N., Hein, M.Y., Min, K.H.J., Wang, L., et al.: Mapping transcriptomic vector fields of single cells. Cell **185**(4), 690–711 (2022)
50. Risso, D., Perraudeau, F., Gribkova, S., Dudoit, S., Vert, J.P.: A general and flexible method for signal extraction from single-cell rna-seq data. Nature communications **9**(1), 284 (2018)
51. Saelens, W., Cannoodt, R., Todorov, H., Saeys, Y.: A comparison of single-cell trajectory inference methods. Nature biotechnology **37**(5), 547–554 (2019)
52. Schiebinger, G., Shu, J., Tabaka, M., Cleary, B., Subramanian, V., Solomon, A., Gould, J., Liu, S., Lin, S., Berube, P., et al.: Optimal-transport analysis of single-cell gene expression identifies developmental trajectories in reprogramming. Cell **176**(4), 928–943 (2019)

53. Sun, T., Song, D., Li, W.V., Li, J.J.: scdesign2: a transparent simulator that generates high-fidelity single-cell gene expression count data with gene correlations captured. Genome biology **22**(1), 163 (2021)
54. Svensson, V., Gayoso, A., Yosef, N., Pachter, L.: Interpretable factor models of single-cell rna-seq via variational autoencoders. Bioinformatics **36**(11), 3418–3421 (2020)
55. Tong, A., Huang, J., Wolf, G., Van Dijk, D., Krishnaswamy, S.: Trajectorynet: A dynamic optimal transport network for modeling cellular dynamics. In: International conference on machine learning. pp. 9526–9536. PMLR (2020)
56. Townes, F.W., Hicks, S.C., Aryee, M.J., Irizarry, R.A.: Feature selection and dimension reduction for single-cell rna-seq based on a multinomial model. Genome biology **20**, 1–16 (2019)
57. Tran, D., Nguyen, H., Tran, B., La Vecchia, C., Luu, H.N., Nguyen, T.: Fast and precise single-cell data analysis using a hierarchical autoencoder. Nature communications **12**(1), 1029 (2021)
58. Trapnell, C., Cacchiarelli, D., Grimsby, J., Pokharel, P., Li, S., Morse, M., Lennon, N.J., Livak, K.J., Mikkelsen, T.S., Rinn, J.L.: The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells. Nature biotechnology **32**(4), 381–386 (2014)
59. Vriz, S., Joly, C., Boulekbache, H., Condamine, H.: Zygotic expression of the zebrafish sox-19, an hmg box-containing gene, suggests an involvement in central nervous system development. Molecular brain research **40**(2), 221–228 (1996)
60. Wang, D., Gu, J.: Vasc: dimension reduction and visualization of single-cell rna-seq data by deep variational autoencoder. Genomics, proteomics & bioinformatics **16**(5), 320–331 (2018)
61. Wang, P., Song, C., Zhang, H., Wu, Z., Tian, X.J., Xing, J.: Epigenetic state network approach for describing cell phenotypic transitions. Interface focus **4**(3), 20130068 (2014)
62. Warga, R.M., Mueller, R.L., Ho, R.K., Kane, D.A.: Zebrafish tbx16 regulates intermediate mesoderm cell fate by attenuating fgf activity. Developmental biology **383**(1), 75–89 (2013)
63. Woicik, A., Zhang, M., Chan, J., Ma, J., Wang, S.: Extrapolating heterogeneous time-series gene expression data using sagittarius. Nature Machine Intelligence pp. 1–15 (2023)
64. Wolf, F.A., Angerer, P., Theis, F.J.: Scanpy: large-scale single-cell gene expression data analysis. Genome biology **19**, 1–5 (2018)
65. Wolf, F.A., Hamey, F.K., Plass, M., Solana, J., Dahlin, J.S., Göttgens, B., Rajewsky, N., Simon, L., Theis, F.J.: Paga: graph abstraction reconciles clustering with trajectory inference through a topology preserving map of single cells. Genome biology **20**, 1–9 (2019)
66. Xiang, R., Wang, W., Yang, L., Wang, S., Xu, C., Chen, X.: A comparison for dimensionality reduction methods of single-cell rna-seq data. Frontiers in genetics **12**, 646936 (2021)
67. Xiong, B., Rui, Y., Zhang, M., Shi, K., Jia, S., Tian, T., Yin, K., Huang, H., Lin, S., Zhao, X., et al.: Tob1 controls dorsal development of zebrafish embryos by antagonizing maternal β-catenin transcriptional activity. Developmental cell **11**(2), 225–238 (2006)
68. Yeo, G.H.T., Saksena, S.D., Gifford, D.K.: Generative modeling of single-cell time series with prescient enables prediction of cell trajectories with interventions. Nature communications **12**(1), 3222 (2021)
69. Zappia, L., Phipson, B., Oshlack, A.: Splatter: simulation of single-cell rna sequencing data. Genome biology **18**(1), 174 (2017)

A Supplementary Notes

A.1 Dataset and Pre-Processing

We use six publicly available datasets. The EB data is downloaded from the Mendeley Data repository at https://data.mendeley.com/datasets/v6n743h5ng/1. MP [3] data is available at https://github.com/theislab/pancreatic-endocrinogenesis. ZB and MB datasets are downloaded from Single Cell Portal with identifier SCP126 and SCP1290 correspondingly. The DR data is downloaded from https://shendure-web.gs.washington.edu/content/members/DEAP_website/public/ and SC data are available at https://broadinstitute.github.io/wot/tutorial/.

For both interpolating and extrapolating tasks of each dataset, we first select highly variable genes (HVGs). HVGs are a subset of genes that contribute strongly to cell-to-cell variation within a cell population. Using only HVGs for analysis helps remove data noise and reduce computational costs. In each case, we detect the top 2000 HVGs only from training timepoints to avoid data leakage (i.e., information from outside the training timepoints is used in the model). For instance, in the EB interpolating task where we use timepoints $t = 0, 1, 3, 4$ to predict expressions at $t = 2$, we select 2000 HVGs based on $t = 0, 1, 3, 4$ in the pre-processing. We use *Scanpy* [64] and *Serurat* [26] to detect HVGs.

We then normalize expression to remove cell-specific bias before conducting experiments. Specifically, given the unique molecular identifier (UMI) count expression of cell i as $\mathbf{X}_i \in \mathbb{Z}^p_{\geq 0}$ where $\mathbb{Z}_{\geq 0} = \{0, 1, 2, \cdots\}$, we normalize it by total counts over all genes

$$\mathbf{X}_i = \frac{\mathbf{X}_i}{\sum_{j=1}^{p} \mathbf{X}_i} * 10^4, \tag{8}$$

followed by the log-transformation

$$\mathbf{X}_{ij} = \log(\mathbf{X}_{ij} + 1). \tag{9}$$

The normalization is cell-specific, such that there is no data leakage. Because MB and SC datasets only provide normalized expression data, we normalize the other four datasets.

A.2 `scNODE` Training and Inference

Our `scNODE` is implemented with *Pytorch 1.13* [47] and is trained end-to-end. `scNODE` training consists of two main steps. `scNODE` first pre-trains the VAE component (encoder Enc_ϕ and decoder Dec_θ) with all cells of training timepoints. We use Adam optimizer to pre-train `scNODE` with a learning rate of 0.001 and 200 iterations. Then, `scNODE` optimizes both VAE and neural ODE components by minimizing the $\mathcal{L}_{\text{dynamic}}$. We adopt batch training and use Adam optimizer to train `scNODE` with a learning rate of 0.001 and 1000 iterations. At each training iteration, we select 32 cells at $t = 0$ as a batch and predict for every $t \in \mathcal{T}$. Because the Wasserstein distance computation is expensive, batch training improves training efficiency and enables `scNODE` usage on large-scale datasets. We use *geomloss* [22] to compute Wasserstein distance with blur = 0.05 and scaling = 0.5. Pseudo-codes of `scNODE` are provided in Algorithm 1.

Specifically, `scNODE` training loss function (Eq. 6) contains a regularization term to pose cellular dynamics to the latent space and combines structure relation and temporal development of cells, leading to an informative and interpretable latent space. Specifically, at each observed timepoint $t \in \mathcal{T}$, we use the encoder to compute latent variables $\mathbf{Z}^{(t)}_{\text{enc}}$ for $\mathbf{X}^{(t)}$ and then compute the Wasserstein distance between it and latent variables $\mathbf{Z}^{(t)}$ from ODE. Notice at the first timepoint $t = 0$, its latent variable $\mathbf{Z}^{(0)}$ is computed from VAE encoding, such that the regularization $\text{Wass}\left(\mathbf{Z}^{(0)}_{\text{enc}},\ \mathbf{Z}^{(0)}\right) \approx 0$. But for the latter measured timepoints, this regularization enforces similarities between structure latent representations from VAE and the dynamic manifold from ODE.

At inference, `scNODE` first uses the trained encoder Enc_ϕ to map gene expression at the first timepoint to the latent space and then samples latent variables for the required number of cells from the latent Gaussian distribution. Then, `scNODE` uses the ODE solver (Eq. 4) to predict latent variables of these cells at required timepoints and map them back to the gene space through the decoder Dec_θ. Our `scNODE` can predict gene expression at any timepoints, including those measured and unmeasured ones. We set step size $\Delta t = 0.1$ for `scNODE` training and inference.

A.3 Baseline Models

We compare `scNODE` with the following four baseline models.

- **PRESCIENT**: [68] proposes a generative model, called Potential eneRgy undErlying Single Cell gradIENTs (PSCIENT), to learn the differentiation landscape from single-cell time-series gene expression data. PRESCIENT maps gene expression to the principal component analysis (PCA) space and models cellular differentiation with neural ordinary differential equation (ODE). PRESCIENT suggests using cell proliferation information to improve prediction. But cell proliferation is not available in general cases. Therefore, we do not use such information in our experiments in all methods. We use Python codes on GitHub (https://github.com/gifford-lab/prescient) to run PRESCIENT.

- **WOT**: Waddintong-OT (WOT) [52] is an approach for studying time-series scRNA-seq data to infer ancestor-descendant fates. WOT can recover a held-out timepoint through linear interpolation based on transport maps constructed between sets of cells in early and late timepoints. Because in extrapolating conditions, the cell-cell transport coupling is unavailable when the subsequent timepoint is missing, WOT can only predict interpolations. We use WOT in interpolating tasks of all datasets. We use Python codes on GitHub (https://github.com/broadinstitute/wot) to run WOT.

- **TrajectoryNet**: [55] utilizes continuous normalizing flow (CNF) [11] to model the paths of cells over time and interpolates between measured timepoints. TrajectoryNet maps gene expression to the PCA space and models cellular dynamics by solving a dynamic optimal transport problem. We use Python codes on GitHub (https://github.com/KrishnaswamyLab/TrajectoryNet) to run TrajectoryNet. Because the published codes can only interpolate one leave-out timepoint, we use TrajectoryNet in predicting EB and MP data.

- **Naive**: We add the Naive model as an intuitive baseline. Specifically, Naive model replicates the true gene expression of the last observed timepoint for prediction. For example, if we predict $t = 3, 4$ from $t = 0, 1, 2$, Naive just use $\mathbf{X}^{(2)}$ as the prediction at $t = 3$ and $t = 4$. The `Naive` model gives costless prediction without learning.

We let every model predict 2000 cells at each testing timepoints to ensure a fair metric comparison. To reason for this choice, We compare method predictions with a different number of cells ranging from 100 to 5000. We find the number of predicted cells has little to no effect on evaluation metrics. Moreover, datasets used in our experiments have thousands of cells at each timepoint, so we think 2000 is a reasonable number.

B More on `scNODE` Prediction

We visualize ground truth and model prediction in the 2D UMAP space. Specifically, in each case, we compute the PCA space for all timepoints of ground truth and compute the 2D UMAP embedding of the first 50 principal components. Then we project each model's predictions to this UMAP space with the PCA and UMAP transformation initialized from ground truth. We use the *umap* package (https://github.com/lmcinnes/umap) and set n_neighbors=50 and min_dist=0.1.

Specially, we test the power of `scNODE` on predicting multiple extrapolations. We train `scNODE` on the first several timepoints and predict expression at rest timepoints. For example, on the ZB dataset which has 12 timepoints, we train `scNODE` on the first 5/6/7/8/9/10 timepoints and predict data at the rest 7/6/5/4/3/2 timepoints. We test `scNODE` on ZB, MB, and DR datasets. Appendix Fig. A1 shows the accuracy of `scNODE` predicting multiple extrapolations.

We also notice that true cells at each timepoint may have different distributions such that the Wasserstein distance is incomparable per timepoint and degradation might be a random outcome. For instance, the Wasserstein distance at $t = 5$ is lower than it at $t = 6$ does not necessarily indicate the model has better predictions at $t = 5$. Therefore, we also show the 2D UMAP of extrapolating predictions of `scNODE` when trained with different number of timepoints (Appendix Fig. A2) to evaluate extrapolations quantitatively. We find that far future predictions cannot align with ground truth, which validates the existence of deterioration.

We show regularization coefficient β is a trade-off between accuracy and interpretation by evaluating `scNODE` performance using different β values. Appendix Fig. A3 shows that a larger β leads to lower Wasserstein distance but less-interpretable latent space such that different differential trajectories are indistinguishable. To measure the interpretation of latent space, we compute the Calinski-Harabasz (CH) index which evaluates cohesion and separation for all cell types in the 2D UMAP space. The UMAP embedding is computed from latent variables from `scNODE` . A high CH value indicates dense and well separated cell groups. We find $\beta = 1$ is a proper setting to obtain both accuracy and interoperability.

Furthermore, we conduct ablation study to test effects of the pre-training phase. Specifically, we compare the accuracy of `scNODE` predictions between using and not using pre-training. The rest hyper-parameter settings remain the same in two cases. We compare the Wasserstein distance and average ℓ_2 and show the CH value in two cases (Appendix Fig. A4). CH values are also computed in the 2D UMAP space.

C More on Perturbation Analysis

We use ZB data to test `scNODE` in perturbation analysis. We train `scNODE` with all timepoints of ZB data and construct the cellular state path in the latent space. The path is constructed with least action path (LAP), which has been used in previous studies [49,48,61] to construct cell fate transitions. The LAP method aims at finding the optimal path between two cell states while minimizing its action and transition time. Specifically, given starting point $\mathbf{X}_0$ and end point $\mathbf{X}_K$, LAP fins a path discretized as a sequence of K points $\mathcal{P} = \{\mathbf{X}_0, \cdots, \mathbf{X}_K\}$. For each segment constrained between $\mathbf{X}_{k-1}$ and $\mathbf{X}_k$, its tangential velocity is defined as $\mathbf{V}_k = \frac{(\mathbf{X}_k - \mathbf{X}_{k-1})}{\Delta}$ where Δ is the timestep taken by cells from $\mathbf{X}_{k-1}$. Therefore, we define the action $\mathcal{S}$ along the path $\mathcal{P}$ as

$$\mathcal{S} = \frac{1}{2}\sum_{k=1}^{K}\Big(\mathbf{V}_k - \mathrm{Drift}_\omega(\widetilde{\mathbf{X}}_k)\Big)^2 \Delta \quad \text{with } \widetilde{\mathbf{X}}_k = \frac{\mathbf{X}_{k-1} + \mathbf{X} - k}{2}. \tag{10}$$

Here, LAP method aims to align the tangential velocity $\mathbf{V}_k$ with the differential velocity $\mathrm{Drift}_\omega(\widetilde{\mathbf{X}}_k)$ learned by `scNODE` , while having the least transition time. Therefore, the optimal path is

$$\widehat{\mathcal{P}} \;=\; \underset{\mathcal{P},\Delta}{\mathrm{argmin}}\,\mathcal{S} \;=\; \underset{\mathcal{P},\Delta}{\mathrm{argmin}}\,\frac{1}{2}\sum_{k=1}^{K}\Big(\mathbf{V}_k - \mathrm{Drift}_\omega(\widetilde{\mathbf{X}}_k)\Big)^2 \Delta. \tag{11}$$

Solving Eq. 11 consists of two iterative steps

(1) Minimize action by fixing path $\mathcal{P}$ and varying the timestep Δ through

$$\widehat{\Delta} = \underset{\Delta}{\mathrm{argmin}}\,\frac{1}{2}\sum_{k=1}^{K}\left(\frac{\mathbf{X}_k - \mathbf{X}_{k-1}}{\Delta} - \mathrm{Drift}_\omega(\widetilde{\mathbf{X}}_k)\right)^2 \Delta. \tag{12}$$

(2) Minimize action by fixing timestep $\widehat{\Delta}$ and varying path $\mathcal{P}$

$$\widehat{\mathcal{P}} = \underset{\mathbf{X}_1,\cdots,\mathbf{X}_{K-1}}{\mathrm{argmin}}\,\frac{1}{2}\sum_{k=1}^{K}\left(\frac{\mathbf{X}_k - \mathbf{X}_{k-1}}{\widehat{\Delta}} - \mathrm{Drift}_\omega(\widetilde{\mathbf{X}}_k)\right)^2 \Delta. \tag{13}$$

The starting ($\mathbf{X}_0$) and end point ($\mathbf{X}_K$) are fixed in the optimization.

We use *scipy.optimize.minimize* to solve these two objective functions. In our experiments, we construct two paths between the first timepoint and two cell populations (PSM and Hindbrain) with $K = 8$. We set the starting point as the center of cells at the first timepoint (t=0) and the endpoint as the center of the cell population. We initialize timestep as $\Delta = 1$ and $\mathcal{P}$ as equally spaced points from the starting to end points.

When finding the differentially expressed (DE) genes, we augment the LAP path with its nearest neighbors. Specifically, assuming $\mathcal{P} = \{\mathbf{X}_0, \cdots, \mathbf{X}_K\}$ is the LAP path from $\mathbf{X}_0$ to $\mathbf{X}_K$, we have only 8 cells on the path which is insufficient for DE detection. Therefore, for each $\mathbf{X}_k \in \mathcal{P}$, we find its nearest neighbors in

order to augment the path. We use *sklearn.neighbors.NearestNeighbors* to search for 10 nearest neighbors. Then we can use *Scanpy* to detect DE genes for the augmented path with the Wilcoxon rank-sum test.

Lastly, we perturb the expression profile of key genes in all cells at the starting timepoint and let `scNODE` predict trajectories for the perturbed gene expression. The perturbations are expected to result in changes in cell fates. Specifically, we overexpress TBX16 (DE of PSM) or SOX3 (DE of Hindbrain) by multiplying their expression values with 100 at $t = 0$ and use `scNODE` to predict cells at the last timepoint. We classify predicted cells with a Random Forest (RF) classifier. We use *sklearn.ensemble.RandomForestClassifier* with default parameters as the classifier and train it with all unperturbed cell expression as well as cell type labels in original datasets.

D More on Cell Trajectories Recovery

We focus on the interpolating task of ZB dataset, which is removing timepoints $t = 2, 4, 6, 8$ and using `scNODE` to predict them back. We apply partition-based graph abstraction (PAGA) [65] to construct cell trajectories. PAGA is a method that computes the topological structure of cell populations and represents population structures in interpretable graphs. Specifically, we use Louvain clustering [6] to cluster cells in the 2D UMAP space and apply PAGA to construct a structural graph of these clusters. We use *Scanpy* for Louvain clustering.

In addition, we use the Ipsen-Mikhailov (IM) distance [30] to quantitatively measure the similarity between the cell trajectory graphs constructed in each case. $\mathrm{IM}(\mathcal{G}_1, \mathcal{G}_2)$ is a graph similarity measurement defined as the square-root difference between the Laplacian spectrum of graphs $\mathcal{G}_1$ and $\mathcal{G}_2$. It ranges from 0 to 1, where 0 indicates maximum similarity between two graph structures and 1 indicates maximum dissimilarity. We use *nedtrd* package [43] to compute the IM distance.

E Supplementary Tables

Table A1. Hyperparameter search space of `scNODE` and baseline methods. "None" means no hidden layers.

Model	Hyperparameters
`scNODE`	$d = 50$, $\lambda = 0$, $\beta = 1$ encoder network size $\in \{\text{None}, [d], [d, d]\}$ decoder network size $\in \{\text{None}, [d], [d, d]\}$ drift network size $\in \{\text{None}, [d], [d, d]\}$
`PRESCIENT`	$d = 50$ # hidden layers $\in \{1, 2, 3\}$ sd $\in [0.0, 1.0]$ tau $\in [0.0, 0.1]$ gradient clip $\in [0.0, 1, 0]$
`WOT`	epsilon $\in \{0.01, 0.05, 0.1, 0.5, 1.0\}$ lambda1 $\in \{0.1, 1, 10\}$, lambda2 $\in \{5, 50, 500\}$, tau=10000, growth iters=3
`TrajectoryNet`	iters=1000, batch size=100 top_k_reg $\in \{0.0, 0.01, 0.05, 0.1, 0.5, 1.0\}$
`Naive`	N/A

Table A2. Wasserstein distance and ℓ_2 distance between left out timepoints and model predictions on the EB dataset. Bold numbers denote the best prediction. "NA" indicates the model is inapplicable in this case.

Model	Interpolation		Extrapolation	
	$t = 2$		$t = 4$	
	Wasserstein	ℓ_2	Wasserstein	ℓ_2
`scNODE`	**354.57**	**30.49**	**435.41**	**33.08**
`PRESCIENT`	794.51	43.46	718.24	41.09
`WOT`	436.71	36.16	NA	NA
`TrajectoryNet`	370.42	32.54	NA	NA
`Naive`	595.28	39.63	636.06	41.51

Table A3. Wasserstein distance and ℓ_2 distance between left out timepoints and model predictions on the MP dataset. Bold numbers denote the best prediction. "NA" indicates the model is inapplicable in this case.

Model	Interpolation		Extrapolation	
	$t = 2$		$t = 3$	
	Wasserstein	ℓ_2	Wasserstein	ℓ_2
`scNODE`	435.30	**36.96**	**320.75**	**32.75**
`PRESCIENT`	919.32	49.29	851.56	45.85
`WOT`	504.18	41.80	NA	NA
`TrajectoryNet`	**415.57**	38.99	NA	NA
`Naive`	690.60	46.60	541.38	41.87

Table A4. Wasserstein distance and ℓ_2 distance between left out timepoints and model predictions on the ZB dataset. Bold numbers denote the best prediction. "NA" indicates the model is inapplicable in this case.

Model	Interpolation						Extrapolation			
	$t = 4$		$t = 6$		$t = 8$		$t = 10$		$t = 11$	
	Wasserstein	ℓ_2	Wasserstein	ℓ_2	Wasserstein	ℓ_2	Wasserstein	ℓ_2	Wasserstein	ℓ_2
scNODE	**434.28**	**32.35**	**390.50**	**31.15**	**468.41**	**34.57**	**607.87**	**38.78**	**685.47**	**40.78**
PRESCIENT	830.88	43.03	662.15	39.17	669.46	40.32	1063.63	49.09	1112.12	49.97
WOT	575.11	38.17	492.84	37.01	565.70	40.64	NA	NA	NA	NA
Naive	775.63	43.88	695.72	42.28	720.77	44.68	887.69	49.06	943.98	50.08

Table A5. Wasserstein distance and ℓ_2 distance between left out timepoints and model predictions on the MB dataset. Bold numbers denote the best prediction. "NA" indicates the model is inapplicable in this case.

Model	Interpolation						Extrapolation					
	$t = 4$		$t = 6$		$t = 8$		$t = 10$		$t = 11$		$t = 12$	
	Wasserstein	ℓ_2	Wasserstein	ℓ_2	Wasserstein	ℓ_2	Wasserstein	ℓ_2	Wasserstein	ℓ_2	Wasserstein	ℓ_2
scNODE	**107.67**	**18.13**	**111.81**	**18.78**	**122.50**	**17.91**	**113.45**	**18.99**	**118.23**	**18.40**	**125.17**	**19.02**
PRESCIENT	188.77	21.95	162.17	21.18	161.35	19.87	145.39	19.81	151.76	19.61	173.50	20.70
WOT	122.46	20.66	131.29	21.03	141.58	20.34	NA	NA	NA	NA	NA	NA
Naive	209.51	24.51	158.26	23.60	184.43	22.90	158.69	23.07	164.77	22.73	177.77	23.23

Table A6. Wasserstein distance and ℓ_2 distance between left out timepoints and model predictions on the DR dataset. Bold numbers denote the best prediction. "NA" indicates the model is inapplicable in this case.

Model	Interpolation						Extrapolation					
	$t = 4$		$t = 6$		$t = 8$		$t = 8$		$t = 9$		$t = 10$	
	Wasserstein	ℓ_2	Wasserstein	ℓ_2	Wasserstein	ℓ_2	Wasserstein	ℓ_2	Wasserstein	ℓ_2	Wasserstein	ℓ_2
scNODE	**354.10**	**29.25**	**412.94**	**31.76**	**486.53**	**34.99**	**547.91**	**36.93**	554.60	37.21	**691.96**	**40.47**
PRESCIENT	447.31	32.09	474.97	33.22	519.20	35.24	583.07	37.20	**536.83**	**36.16**	692.53	39.80
WOT	426.70	34.35	508.96	37.06	592.67	40.52	NA	NA	NA	NA	NA	NA
Naive	529.27	38.54	607.85	41.14	760.64	45.58	824.57	47.40	824.57	46.90	969.15	49.50

Table A7. Wasserstein distance and ℓ_2 distance between left out timepoints and model predictions on the SC dataset. Bold numbers denote the best prediction. "NA" indicates the model is inapplicable in this case.

Model	Interpolation						Extrapolation					
	$t = 15$		$t = 20$		$t = 25$		$t = 36$		$t = 37$		$t = 38$	
	Wasserstein	ℓ_2	Wasserstein	ℓ_2	Wasserstein	ℓ_2	Wasserstein	ℓ_2	Wasserstein	ℓ_2	Wasserstein	ℓ_2
scNODE	**59.33**	**12.89**	**92.40**	**16.28**	**134.75**	**20.67**	**120.55**	**20.59**	**131.29**	21.19	**127.64**	**20.43**
PRESCIENT	106.14	16.37	129.83	18.60	143.94	20.91	134.32	20.38	142.13	**20.99**	140.70	20.60
WOT	71.36	14.68	115.90	18.79	143.06	23.66	NA	NA	NA	NA	NA	NA
Naive	80.40	15.53	140.79	20.08	197.59	26.15	157.28	24.91	165.89	25.42	165.90	25.03

Table A8. Performance of scNODE predictions using different latent space size (d).

Latent Size (d)	**25**	**50**	**75**	**100**	**125**	**150**	**175**	**200**	**Avg.**	**Std.**
Wasserstein	493.15	453.71	463.31	465.10	463.96	452.79	452.33	453.75	462.26	12.74
ℓ_2	32.10	31.63	31.89	32.08	31.94	32.04	32.02	31.94	31.96	0.14

F Supplementary Figures

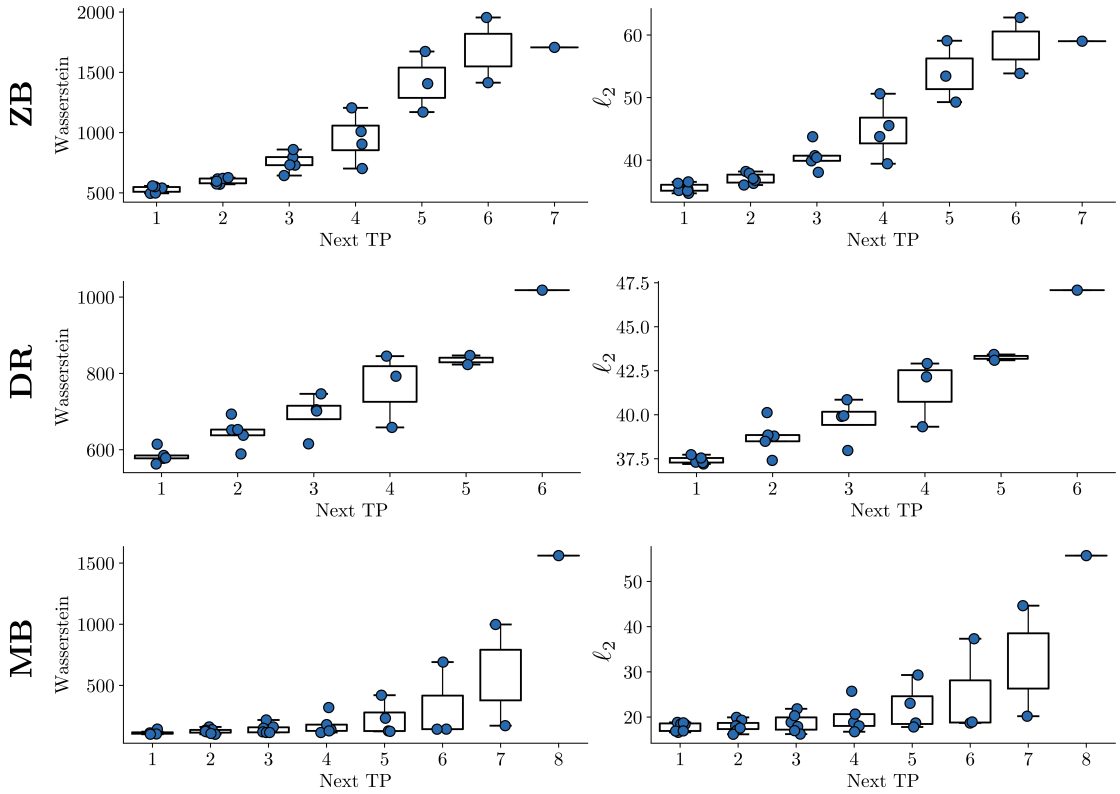

Fig. A1. The Wasserstein distance between true and `scNODE` predictions when using different numbers of training timepoints and predicting extrapolations.

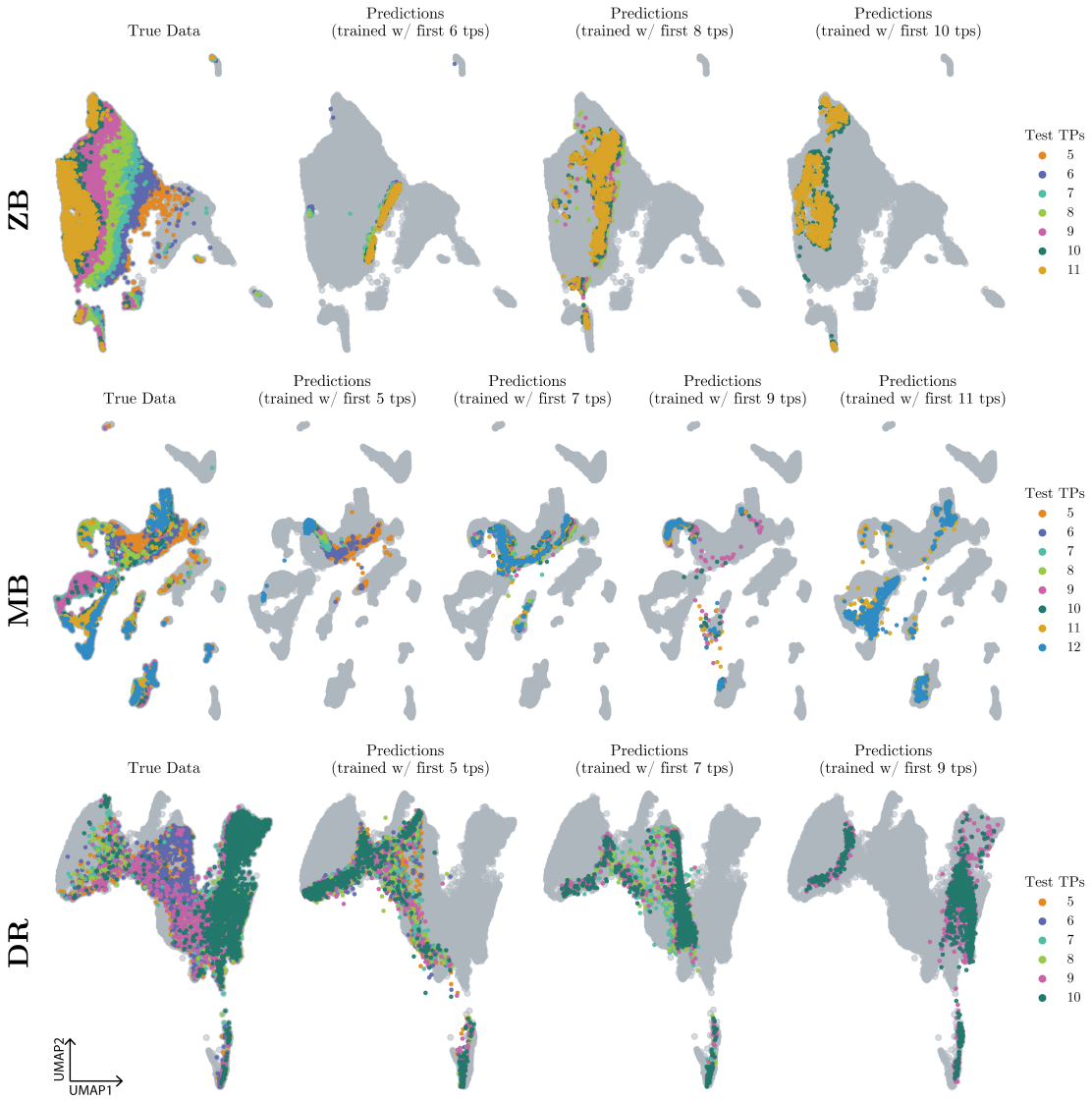

Fig. A2. UMAP visualization of true and `scNODE` predictions when using different numbers of training timepoints.

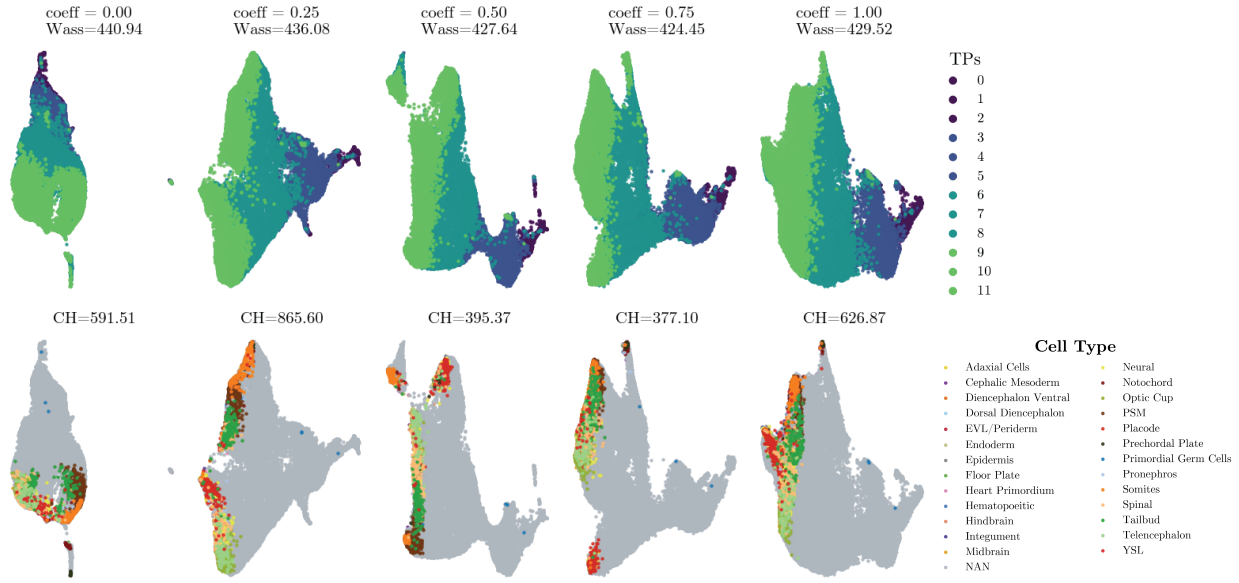

Fig. A3. UMAP visualization of `scNODE` latent space using different coefficient β. The Calinski-Harabasz (CH) index evaluates cohesion and separation for all cell types in the latent space where a high CH value indicates dense and well separated cell groups.

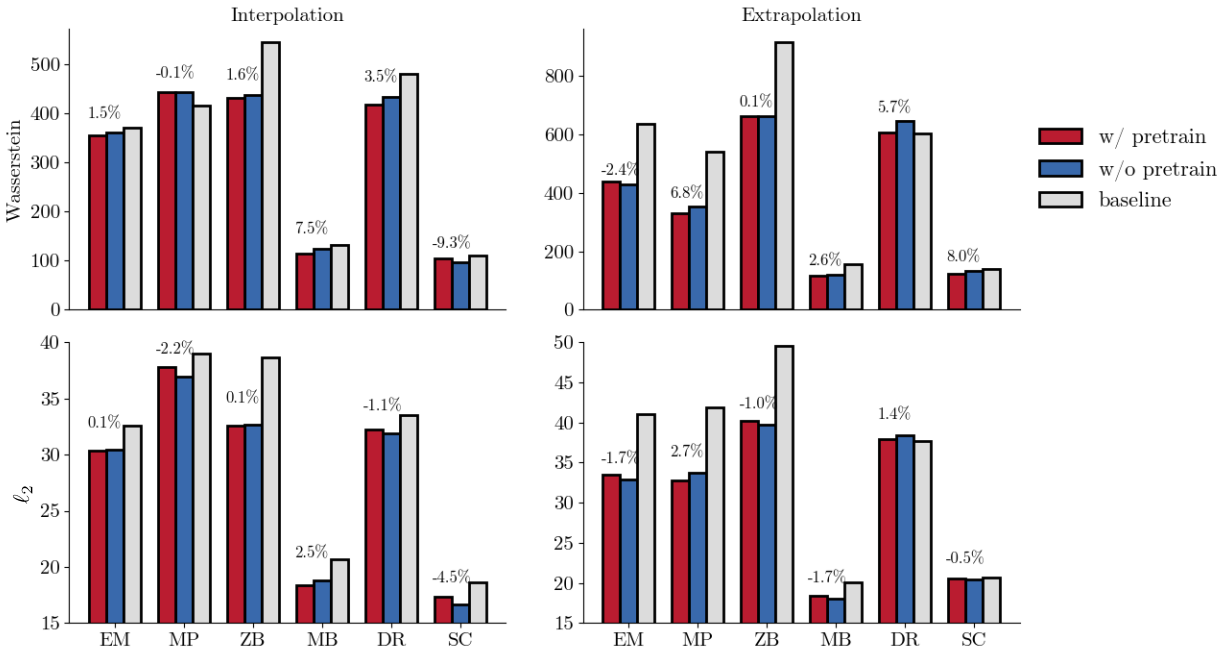

Fig. A4. Comparison of the accuracy for `scNODE` predictions with and without the pre-training phase. Moreover, "baseline" indicates the best performance of all baseline models. Numbers represent the percentage increase or decrease of including the pre-training relative to excluding it.

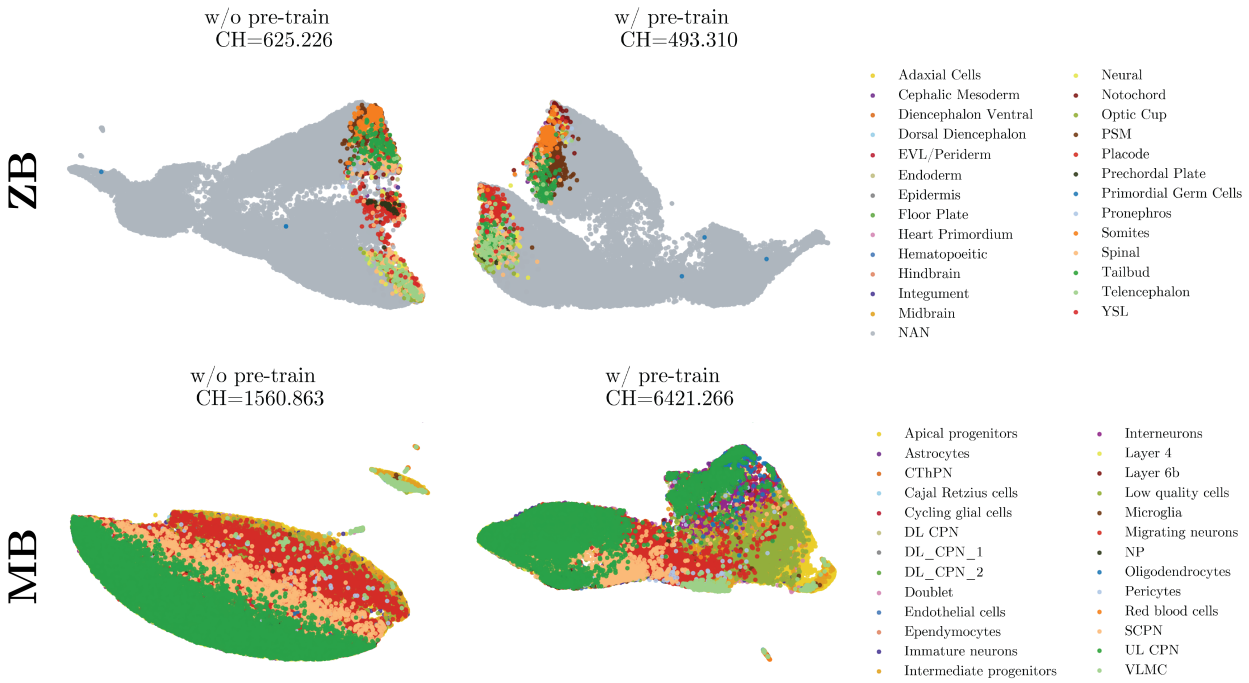

Fig. A5. UMAP visualization of predictions of `scNODE` with and without the pre-training phase on ZB and MB datasets. The Calinski-Harabasz (CH) index evaluates cohesion and separation for all cell types in the latent space where a high CH value indicates dense and well separated cell groups.

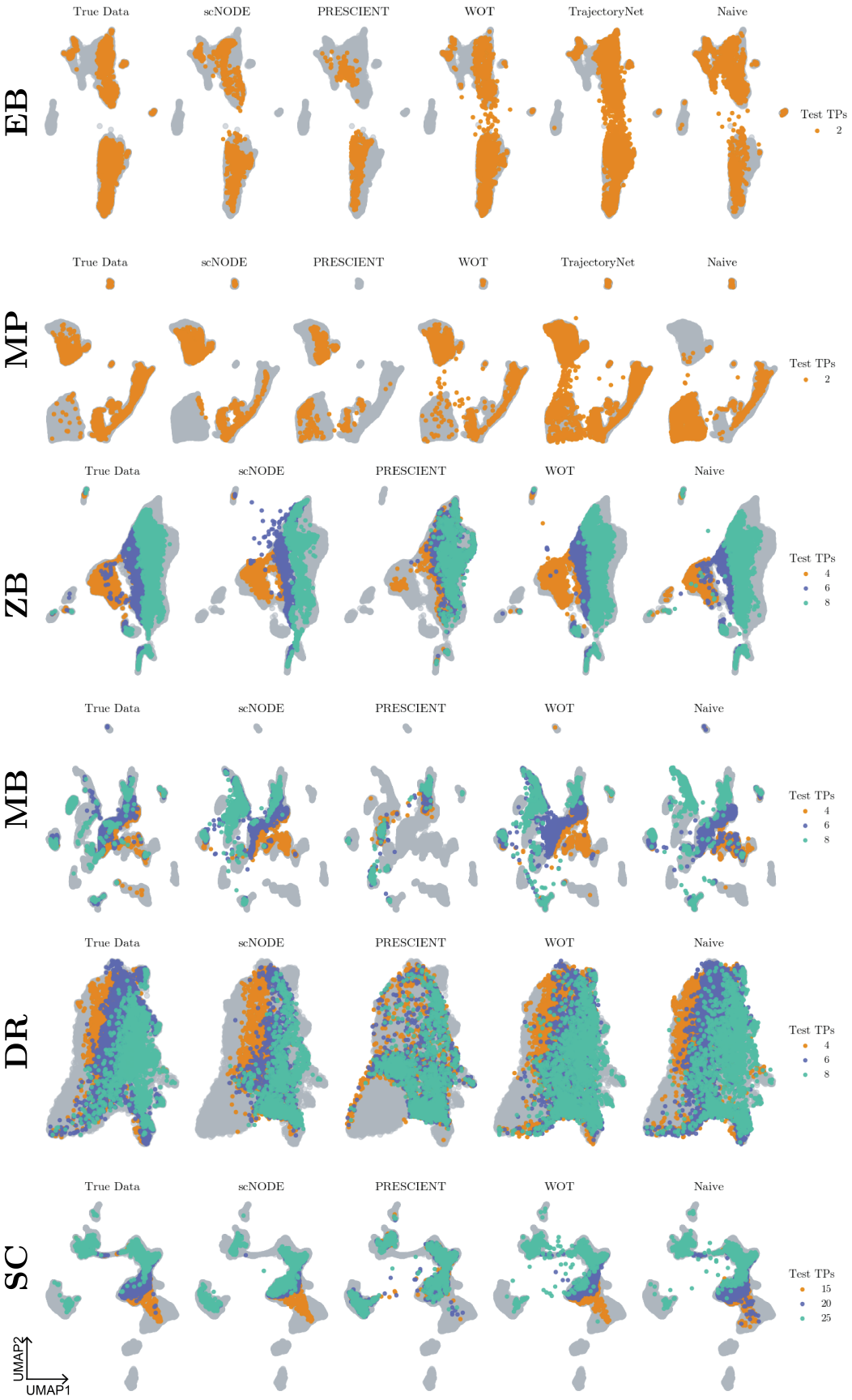

Fig. A6. UMAP visualizations of true and predicted data for the interpolating task. `TrajectoryNet` is inapplicable for interpolating more than one timepoint.

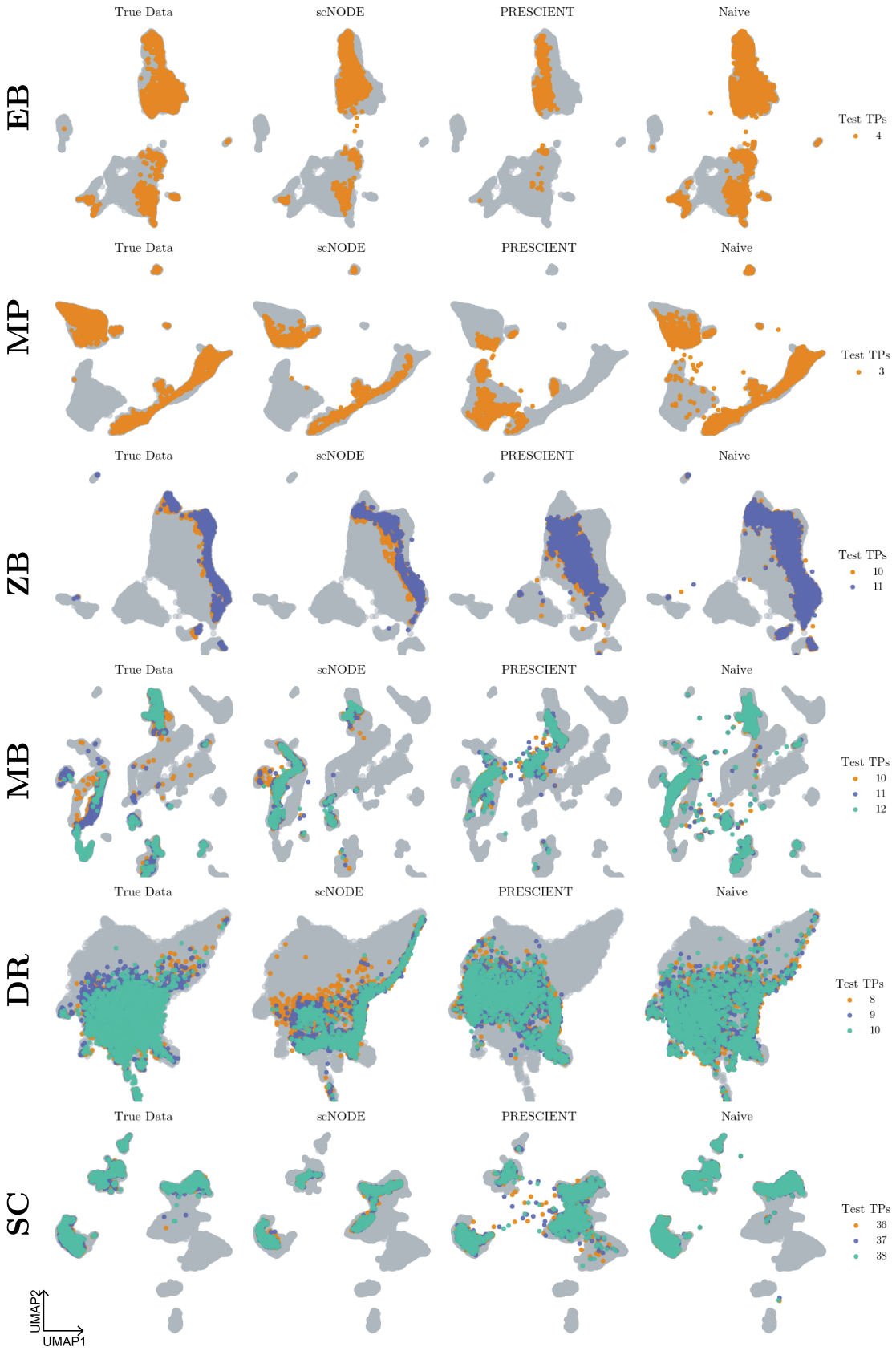

Fig. A7. UMAP visualizations of true and predicted data for the extrapolating task. `Trajectory` and `WOT` are inapplicable for extrapolating.

Fig. A8. `scNODE` perturbation analysis results. (**A**) 2D UMAP visualization of the least action path between cells at the starting point ($t = 0$) and PSM/Hindbrain cell populations. (**B**) Gene expression value of two DE genes (SOX3 and SOX19A) of the Hindbrain path. (**C**) The ratio of Hindbrain cells and other cell types in unperturbed data and predictions from SOX3 perturbations.